Cursul nr. 10

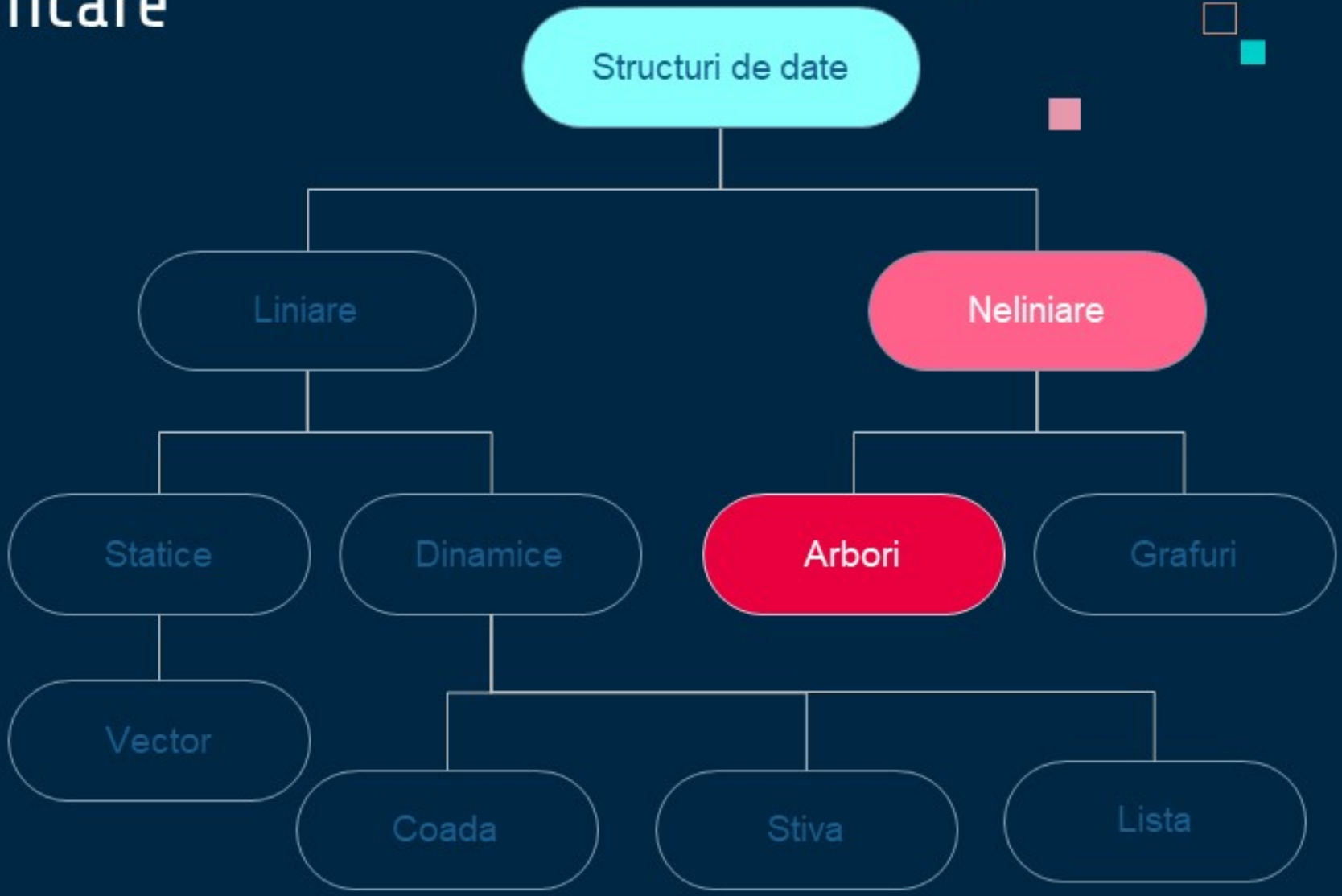
STRUCTURI DE DATE

neliniare

Arbore-B (B-Tree)

Lector dr. Dorin IORDACHE

Clasificare



Agenda

01

Proprietăți

02

Parcurgeri

03

Operații

Proprietăți

01

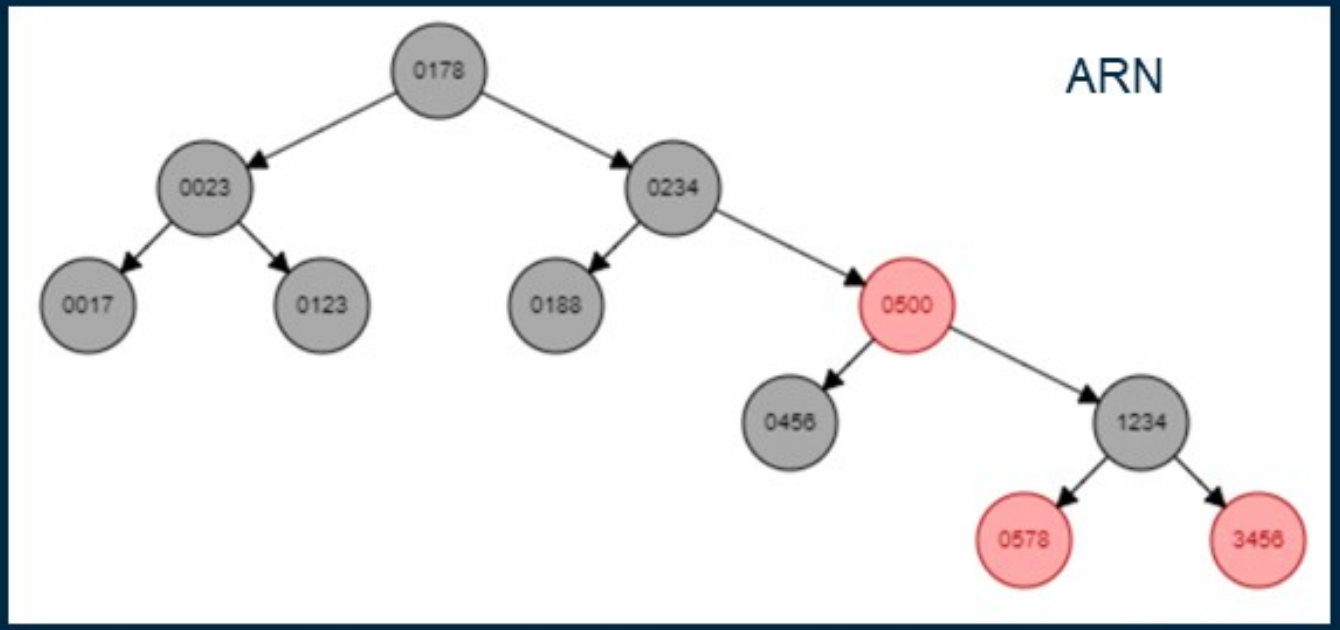
Arbori -B – definire

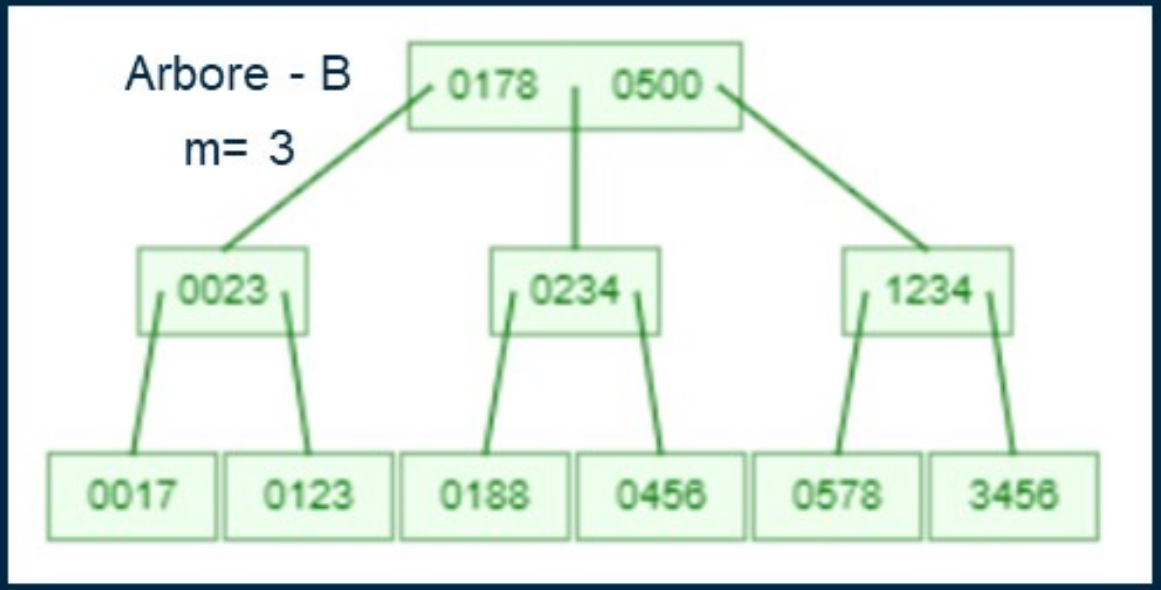
Arbore B = este un ABC cu auto echilibrare

pentru un *m* numar natural nenul dat arbitrar, spunem ca un arbore *m*-ar (cu proprietatea ca orice nod are cel mult *m* fii)

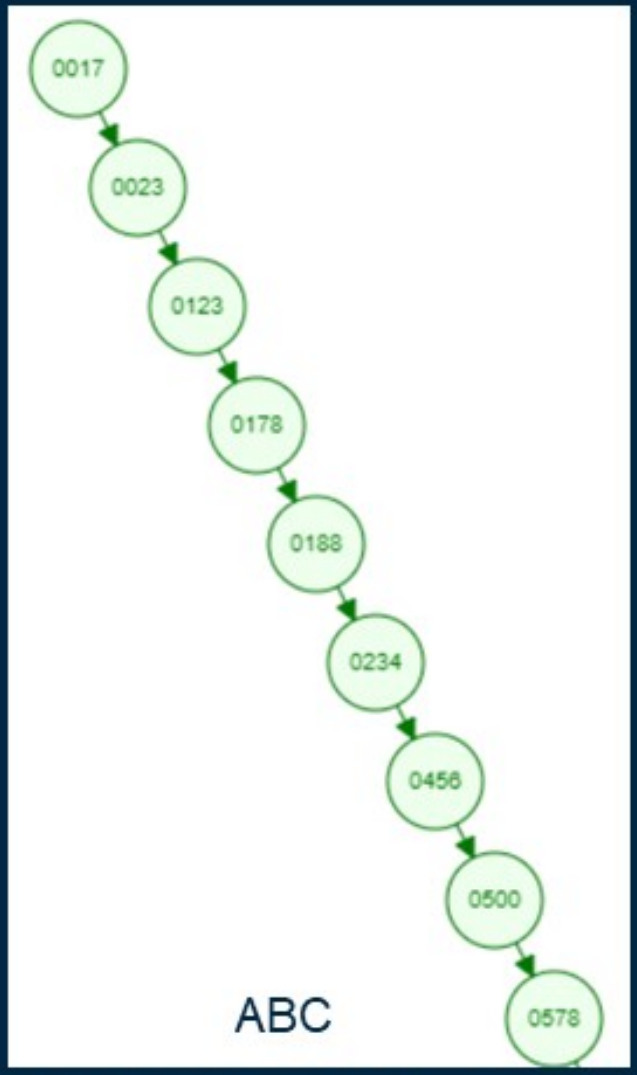
B-Tree îndeplinește următoarele proprietăți:

1. pentru fiecare nod, daca t este numarul de fii, atunci nodul are t - 1 chei;
2. in fiecare nod cheile sunt asezate in ordine crescatoare si au rol de a "separa" fiii;
3. fiul cel mai din stanga al unui nod are toate cheile mai mici decat prima cheie a parintelui; fiul aflat intre doua chei k1 si k2 are cheile cu valori cuprinse in intervalul [k1; k2]; fiul cel mai din dreapta are cheile mai mari decat ultima cheie a parintelui;
4. toate frunzele sunt la acelasi nivel;
5. toate nodurile interne cu exceptia radacinii au cel putin [m/2] fii nevizi;
6. fiecare frunza trebuie sa contina cel putin [m/2]-1 chei si maxim m-1 chei.





Diferențe ?

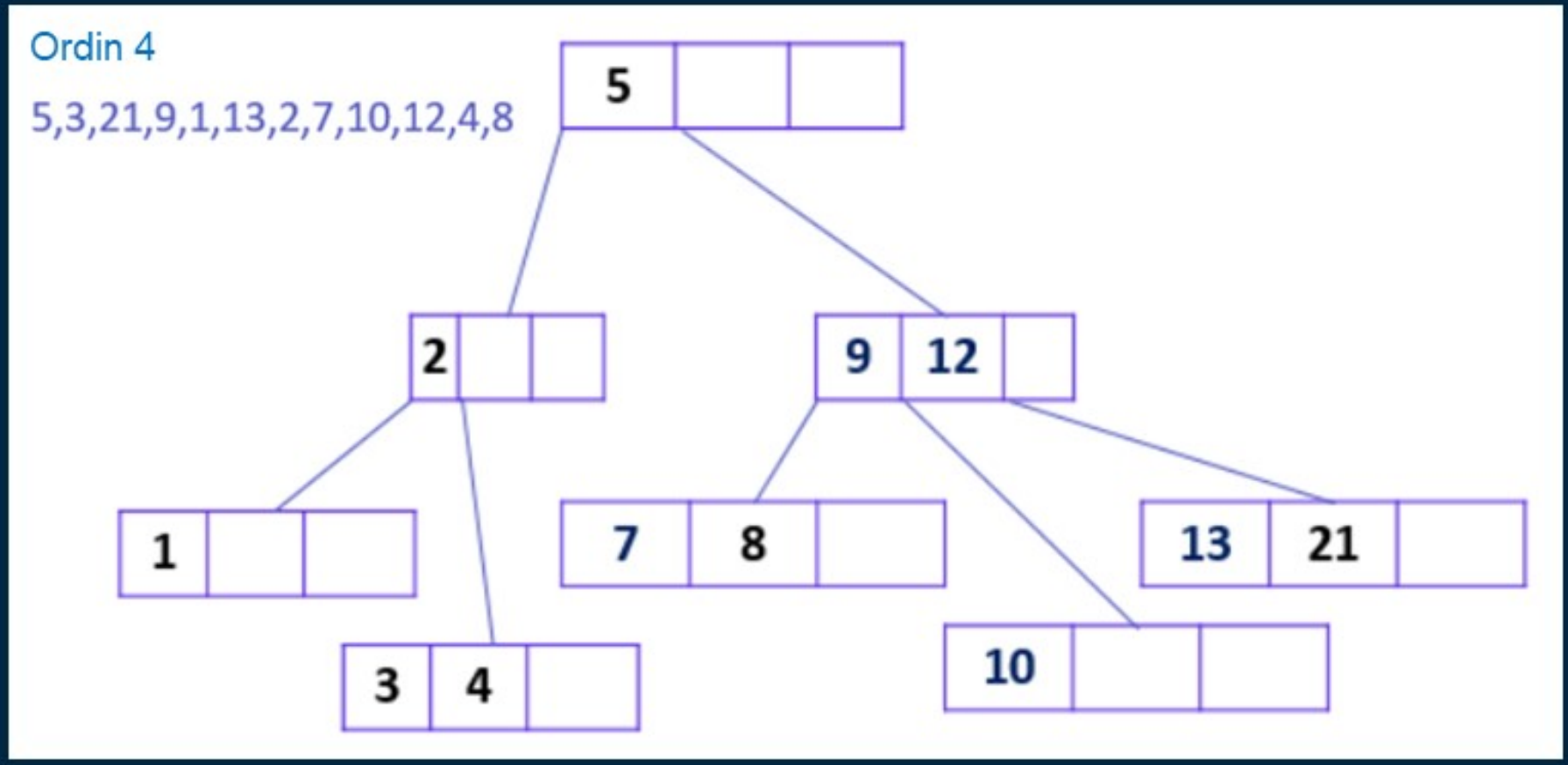


17,23,123,178,188,234,456,500,578,1234,3456

Diferențe

Criteriu	Arbori-B	Arbore binar
Esențial	Un nod poate avea la numărul maxim de M de noduri copil (unde M este ordinea arborelui).	Un nod poate avea cel mult 2 fii.
Folosit	Se utilizează atunci când datele sunt stocate pe disc.	Se utilizează atunci când înregistrările și datele sunt stocate în memoria RAM.
Înălțimea arborelui	$\log_M N$ (unde M este ordinul arborelui M-ar)	$\log_2 N$
Aplicații	Codificarea structurii datelor în multe DBMS.	Optimizarea codului, codificarea lui Huffman etc.

Arbori-B – definire



Parcurgeri

02

Parcurgerea Arbori-B

Efectuarea oricarei operații pe un arbore, necesita accesarea unui nod specific.

Algoritmul de traversare a arborelui – solutia

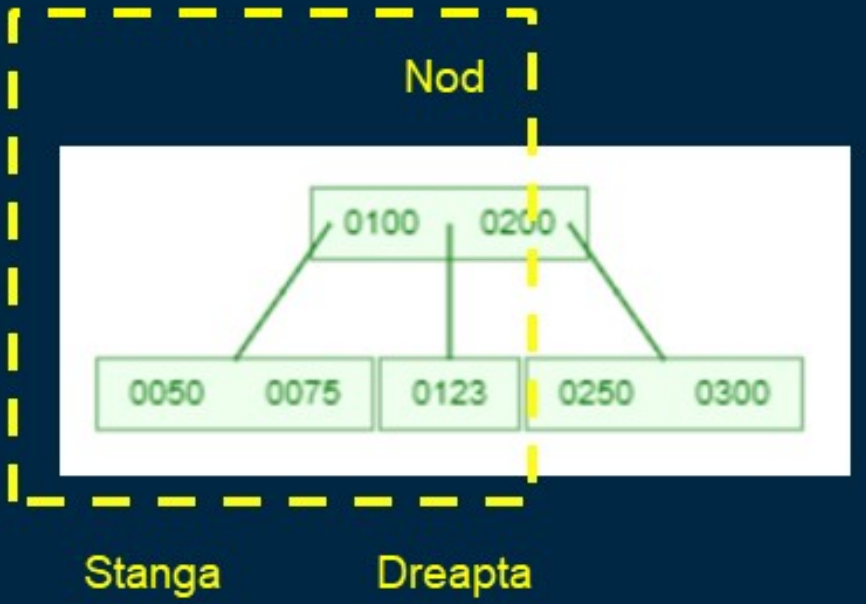
Tipuri de parcurgeri:

- InOrdine
- PreOrdine
- PostOrdine

InOrdine (SND, LNR)

Pas 1: Se parcurg toate nodurile din subarborele din stânga
Pas 2: Apoi nodul rădăcină
Pas 3: Se parcurg toate nodurile din subarborele din dreapta

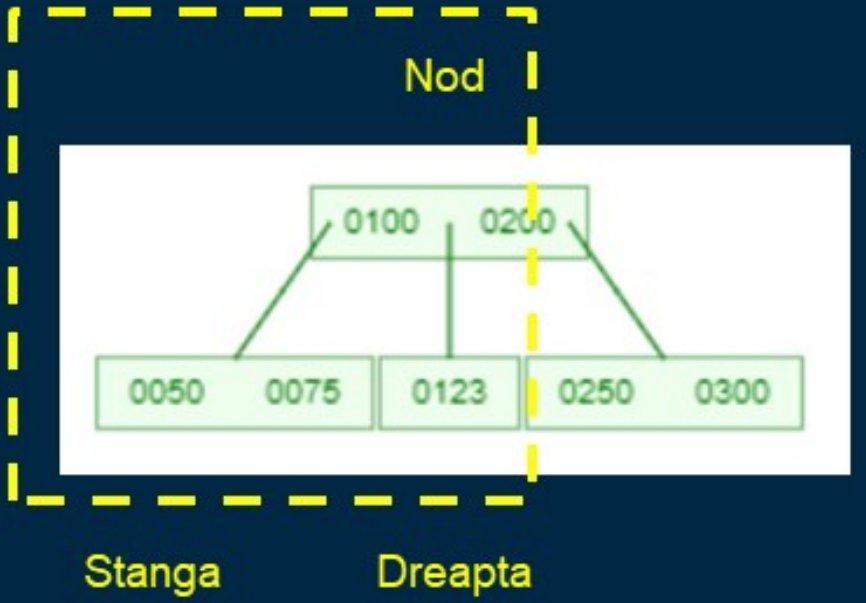
```
InOrdine(root->stanga)
afisare(root->data)
InOrdine(root->dreapta)
```



PreOrdine (NSD, NLR)

Pas 1: Se viziteaza nodul radacina
Pas 2: Se parcurg toate nodurile din subarborele din stânga
Pas 3: Se parcurg toate nodurile din subarborele din dreapta

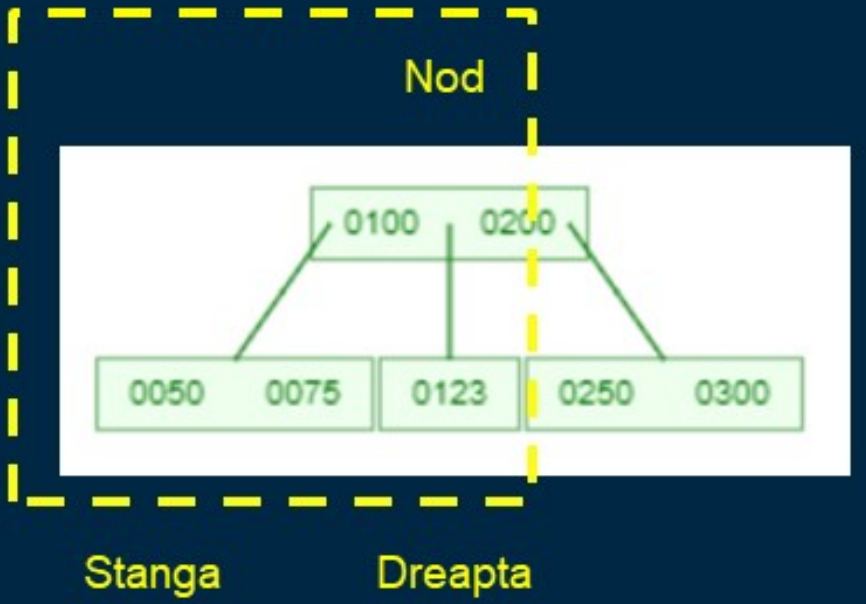
```
afisare(root->data)
PreOrdine(root->stanga)
PreOrdine(root->dreapta)
```



PostOrdine (SDN, LRN)

Pas 1: Se parcurg toate nodurile din subarborele din stânga
Pas 2: Se parcurg toate nodurile din subarborele din dreapta
Pas 3: Se viziteaza nodul radacina

```
PostOrdine(root->stanga)
PostOrdine(root->dreapta)
afisare(root->data)
```



Operații – Arbori-B

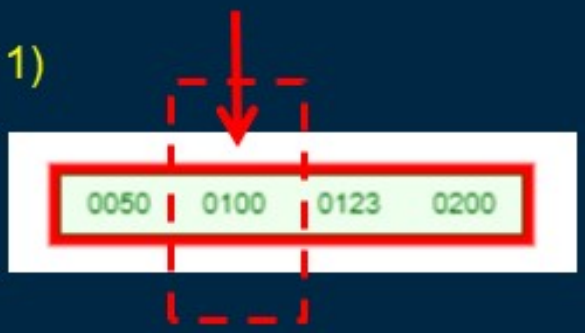
03

Echilibrare Arbori-B

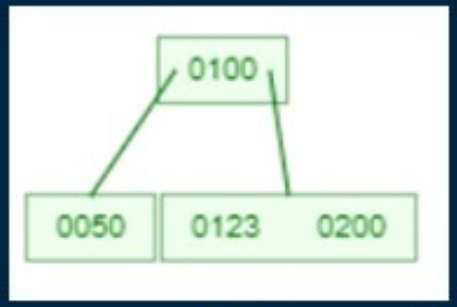
de ordin 4

Split nod

1)

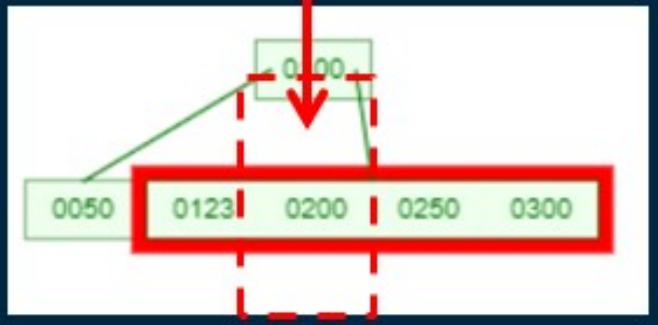


2)

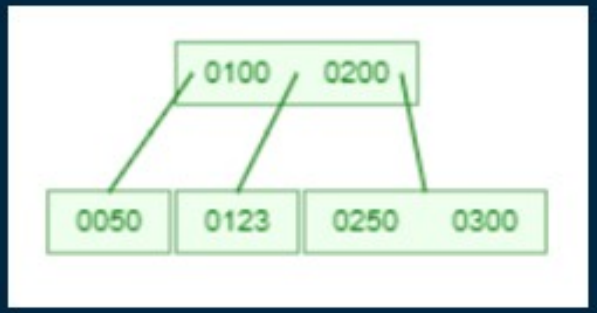


3)

Split nod

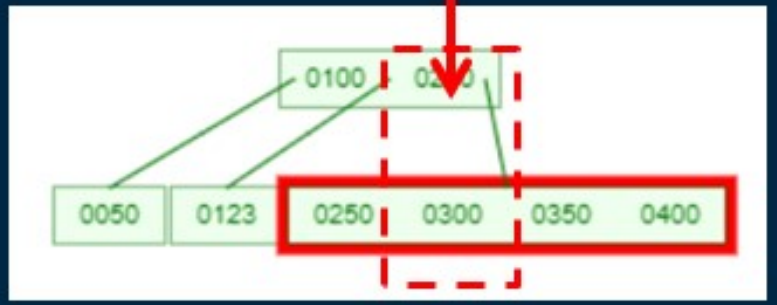


4)



5)

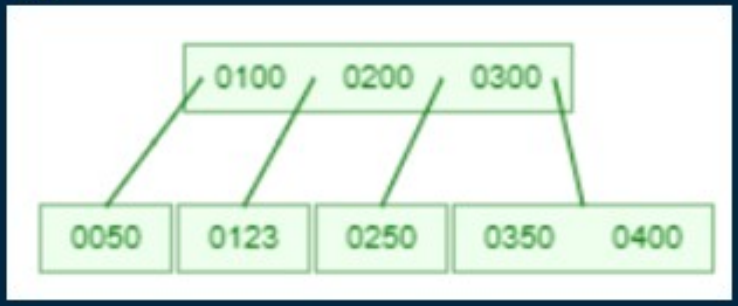
Split nod



6)

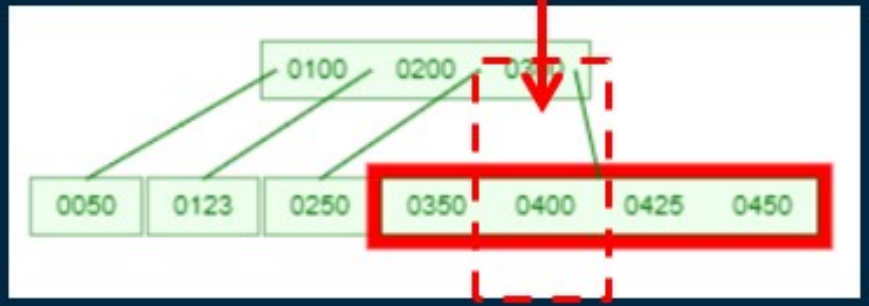
Echilibrare Arbori-B de ordin 4

6)



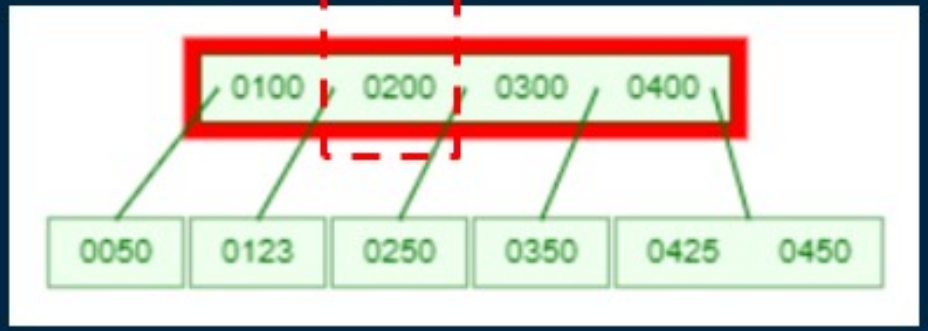
7)

Split nod

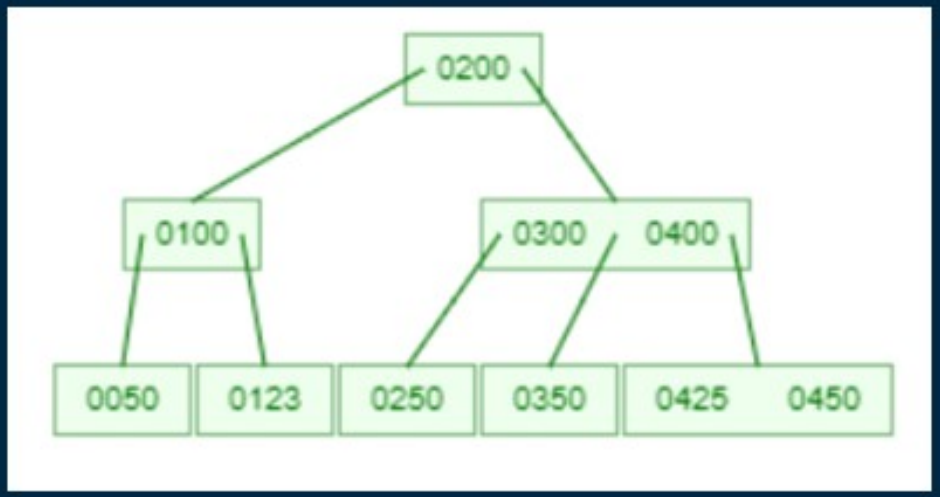


8)

Split nod



final)



Operații Arbori-B

- Căutare
- Adăugare
- Ștergere

Operații Arbori-B

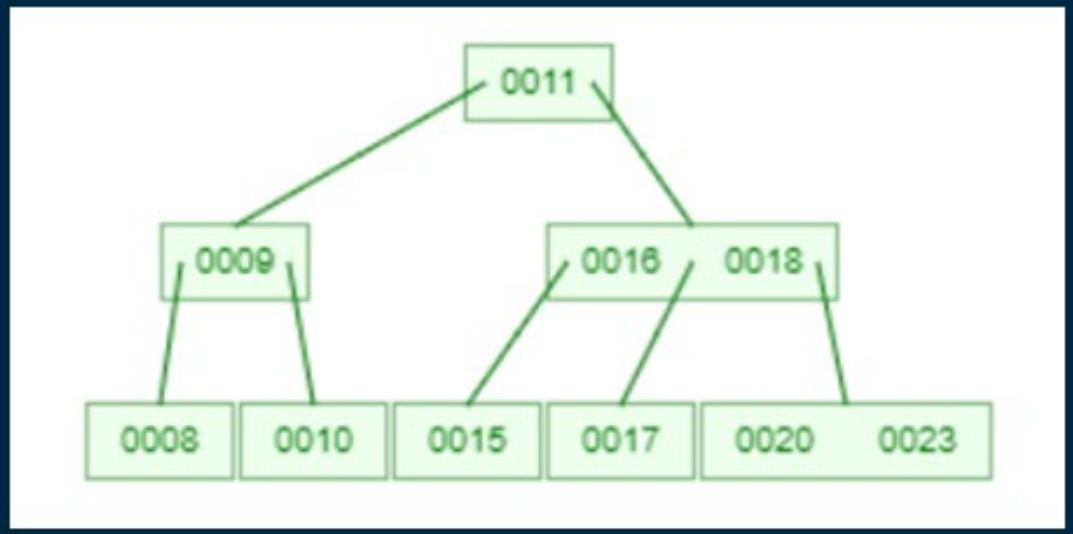
- Căutare = forma generalizată de căutare a unui element într-un arbore binar de căutare. Căutăm nodul cu cheia K

Pas 1: De la nodul rădăcină, comparăm K cu prima cheie a nodului.
Dacă
K = prima cheie a nodului, returnează nodul și indexul.
altfel
returnează NULL (adică nu a fost găsit).
Pas 2: Dacă K < prima cheie a nodului rădăcină, căutăm recursiv copilul din stânga acestei chei.
Pas 3: Dacă există mai mult de o cheie în nodul curent și K > prima cheie, comparăm K cu următoarea cheie din nod.
Dacă K < următoarea cheie, căutăm în nodul copil din stânga acestei chei (adică K se află între prima și a doua cheie).
În caz contrar, căutăm în nodul copil din dreapta cheii.
Pas 4: Se repetă pașii de la 1 la 3 până când se ajunge la frunza (nu mai sunt noduri)

Operații Arbori-B

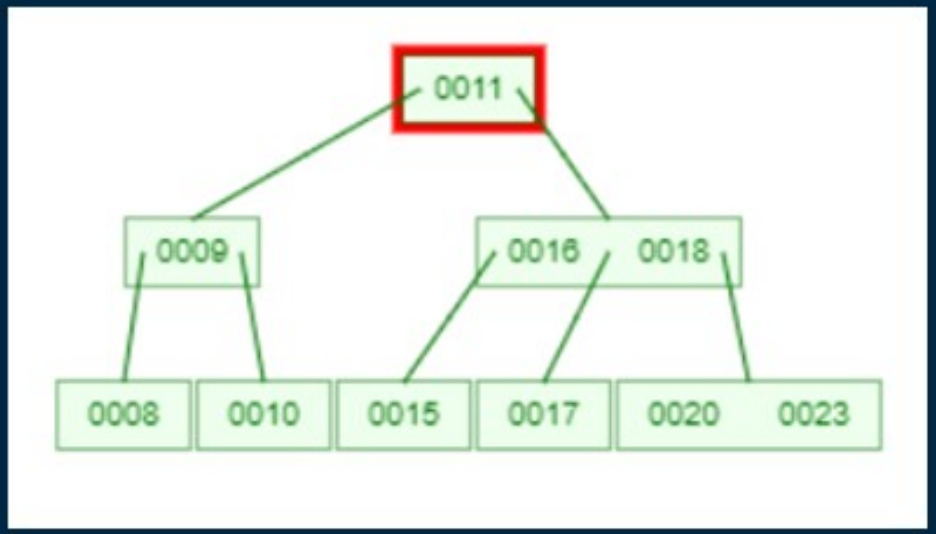
- Căutare

Arbore-B de ordin 3



Căutăm nodul cu cheie 17

1) K=17 > 11

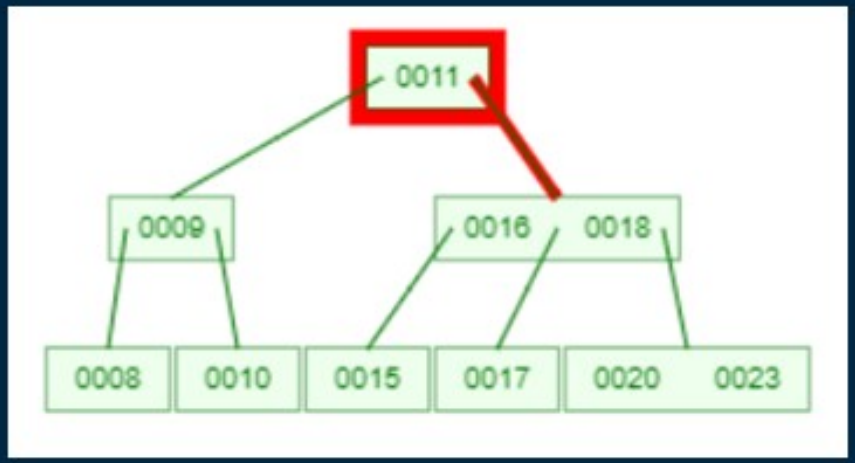


Operații Arbori-B

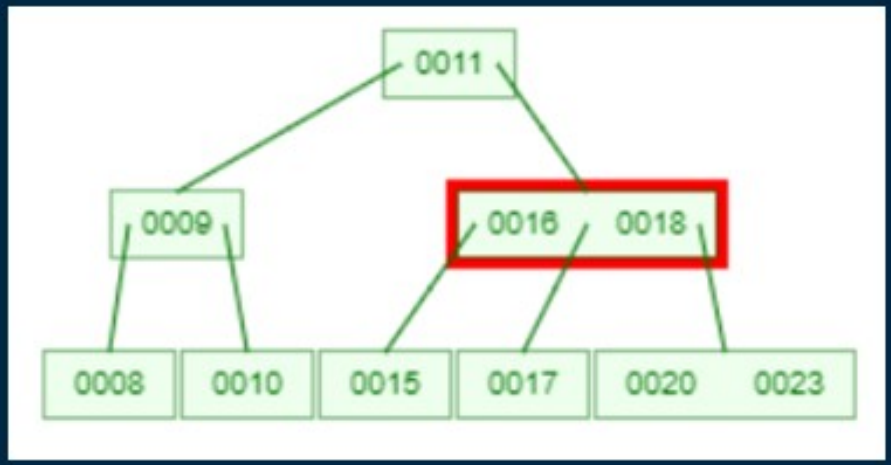
- Căutare

Căutăm nodul cu cheie 17

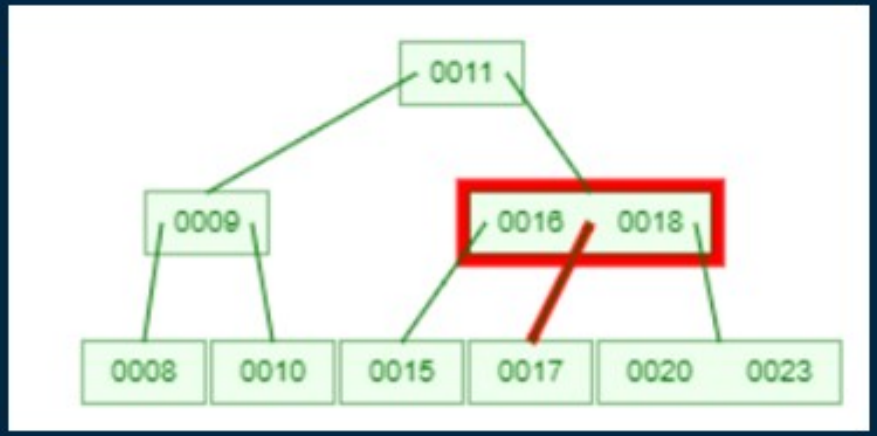
1) K=17 > 11



3) 15 ? K=17 ? 18



3) 15 < K=17 <18

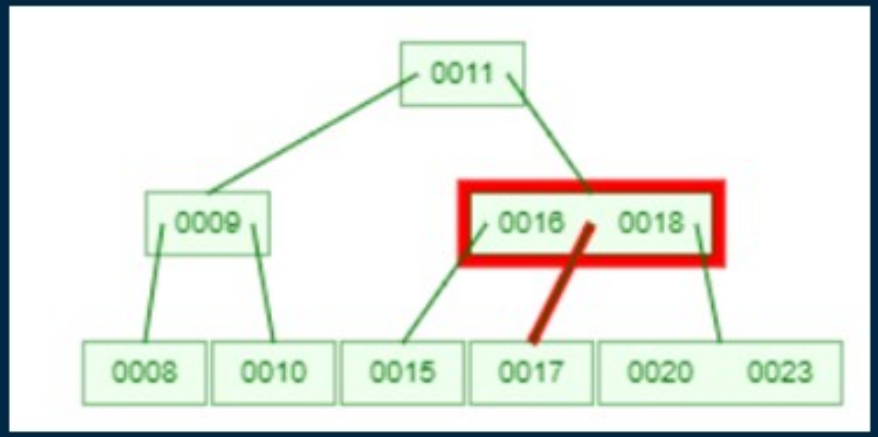


Operații Arbori-B

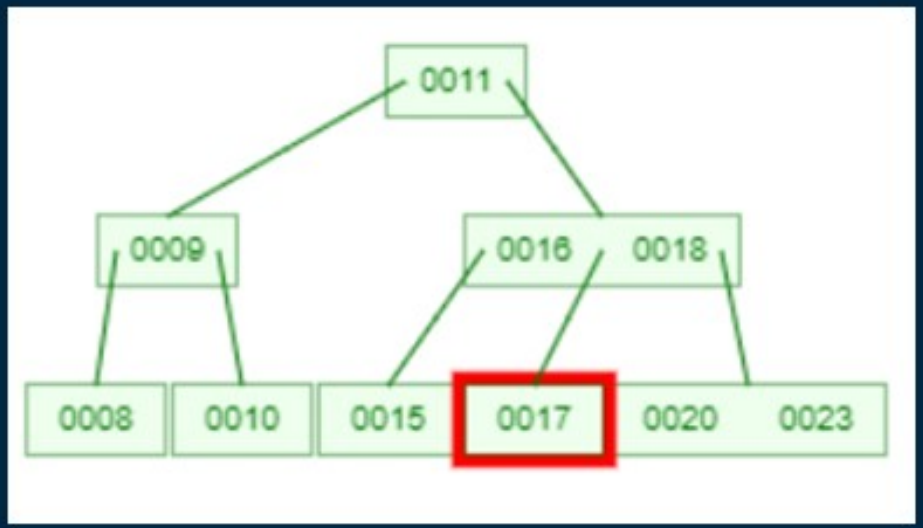
- Căutare

Căutăm nodul cu cheie 17

3) 15 < K=17 <18



final) K=17 true



Operații Arbori-B

- Căutare

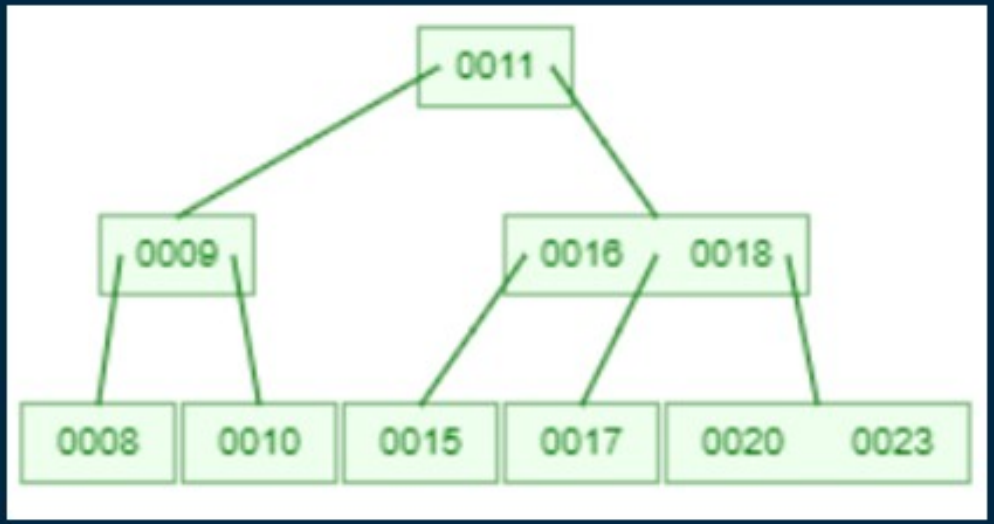
```
begin procedure cautBArbore(x, k)
        i = 1
        while i ≤ n[x] and k ≥ keyi[x] do // n[x] nr chei in nod x
                i = i + 1
                if i=n[x] and k = keyi[x] then
                        return (x, i)
                endif
                if frunza[x] then
                        return NIL
                else
                        return cautBArbore(x, k)
                endif
        endwhile
 end procedure cautBArbore(x, k)
```

Operații Arbori-B

- Adăugare

(1) Dacă arborele este gol, se alocă nod rădăcină.
(2) Se actualizează numărul permis de chei în nod.
(3) Se caută nodul potrivit pentru inserare.
(4) Dacă nodul este plin, se execută pașii următori:
(5) Introduceți elementele în ordine crescătoare.
(6) Există elemente mai mari decât limita sa. Deci, se împaarte nodul la mediană.
(7) Promovați cheia mediană în sus și faceți cheile din stânga noduri copil stâng și cheile din dreapta noduri copil drept.
(8) Dacă nodul nu este plin, se introduce nodul în ordine crescătoare.

Arbore-B

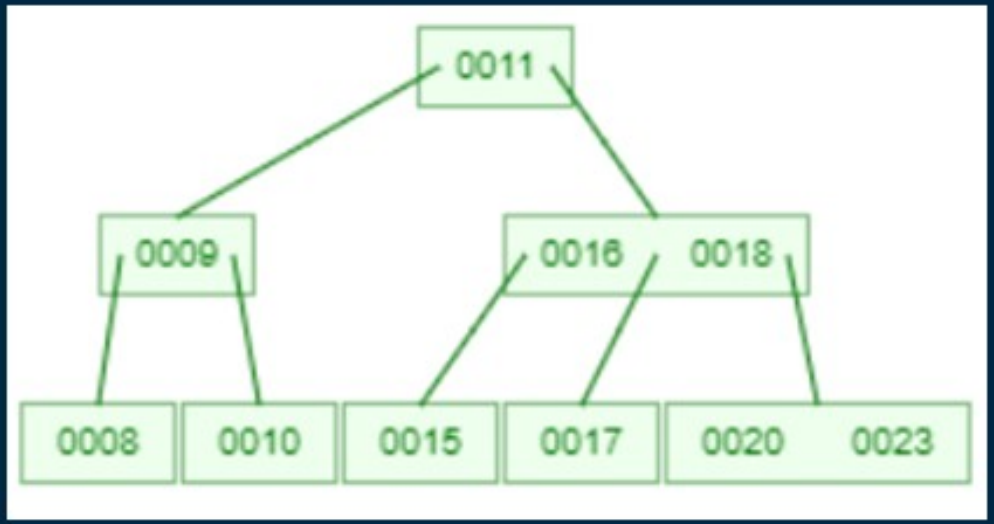


Operații Arbori-B

- Adăugare

Exercițiu:
adăugăm valoarea 121

Arbore-B

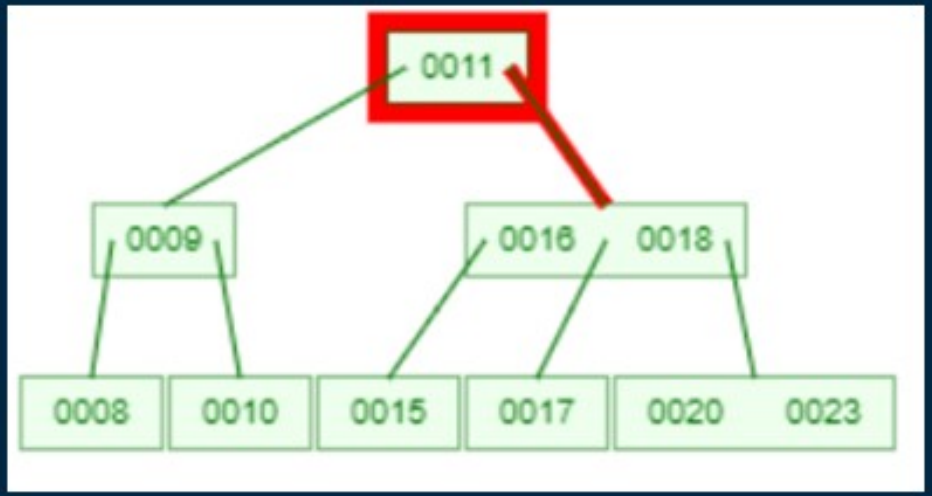


Operații Arbori-B

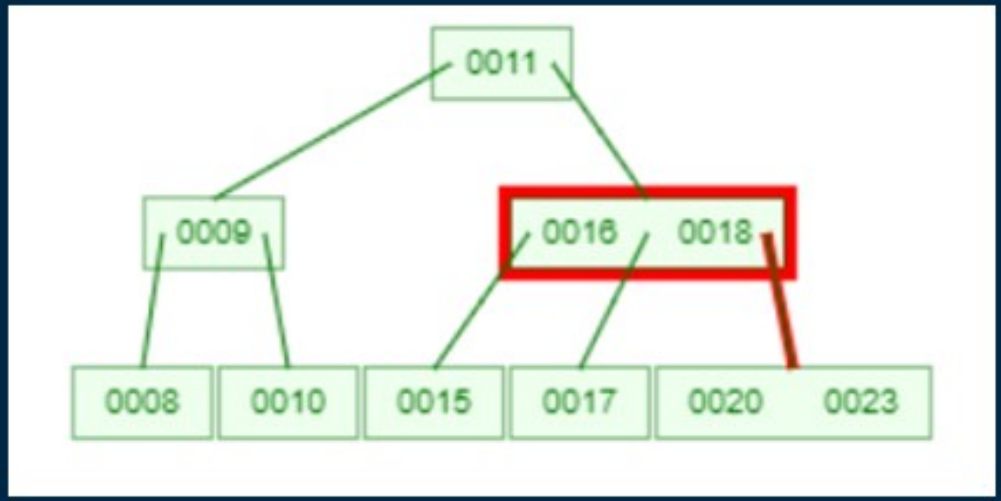
- Adăugare

Exercițiu:
adăugăm valoarea 121

1) K=121 > 11



2) 16 < K=121 < 18

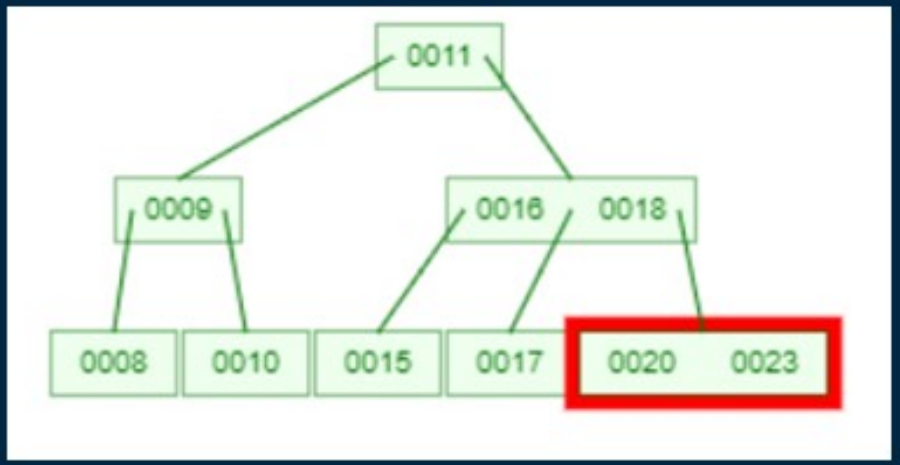


Operații Arbori-B

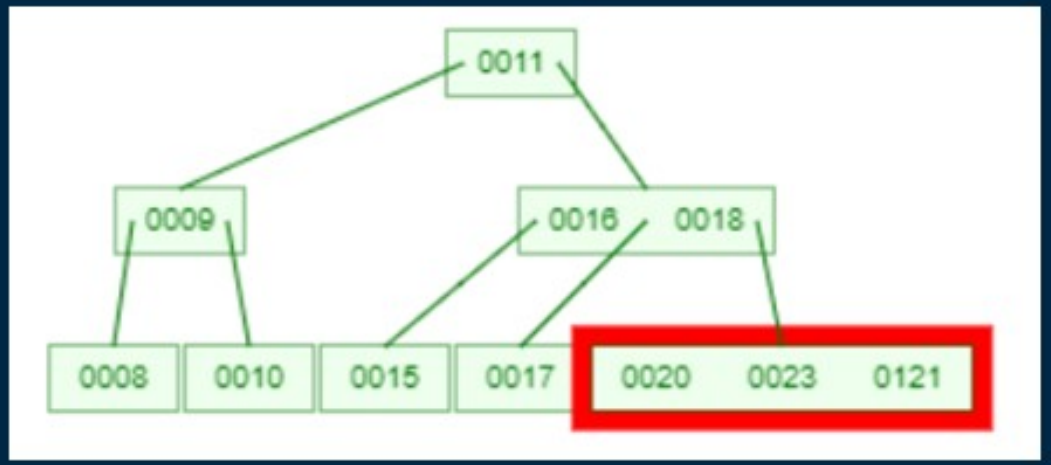
- Adăugare

Exercițiu:
adăugăm valoarea 121

3) 16 < K=121 < 18



4) 20 < 23 < K=121

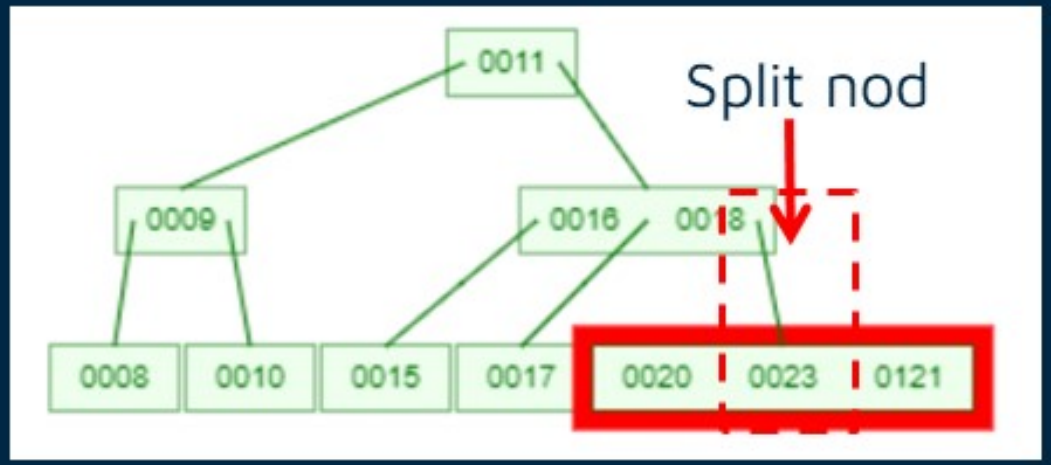


Operații Arbori-B

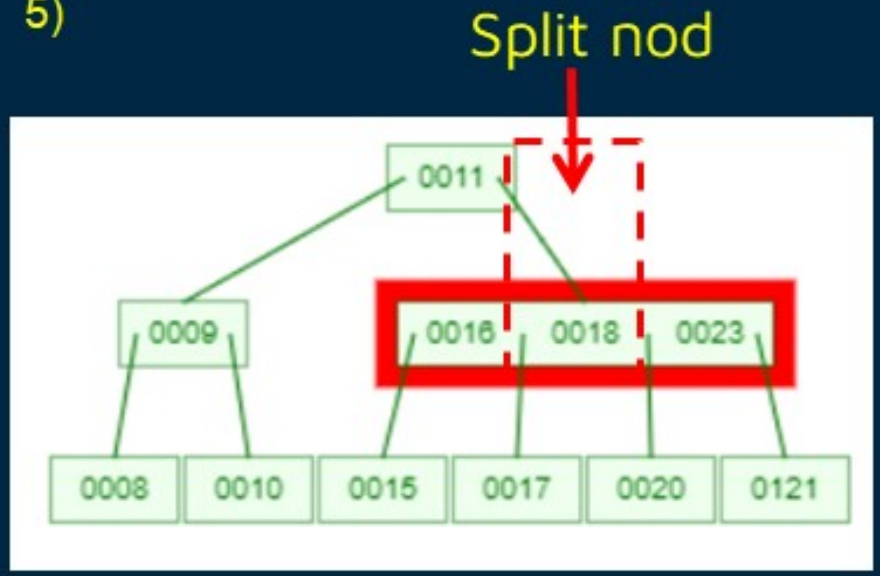
- Adăugare

Exercițiu:
adăugăm valoarea 121

4) 20 < 23 < K=121



5)

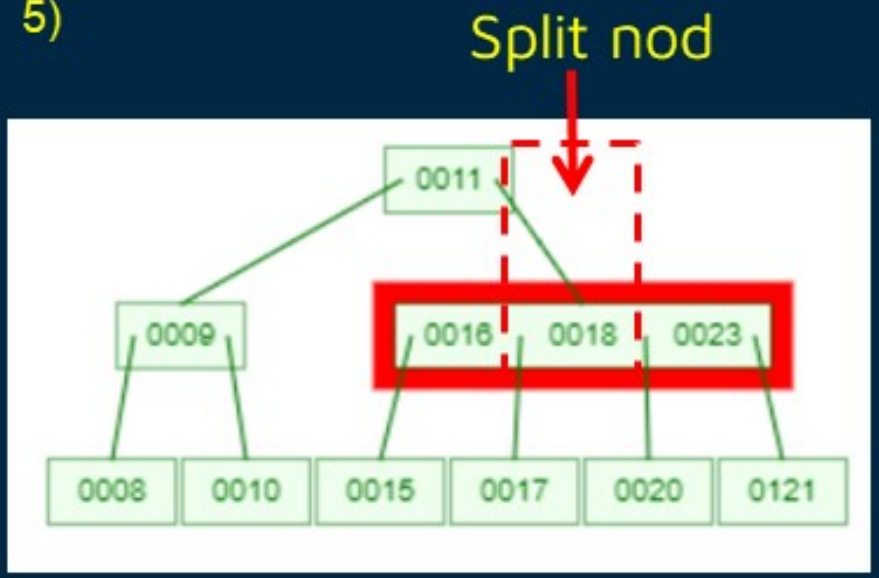


Operații Arbori-B

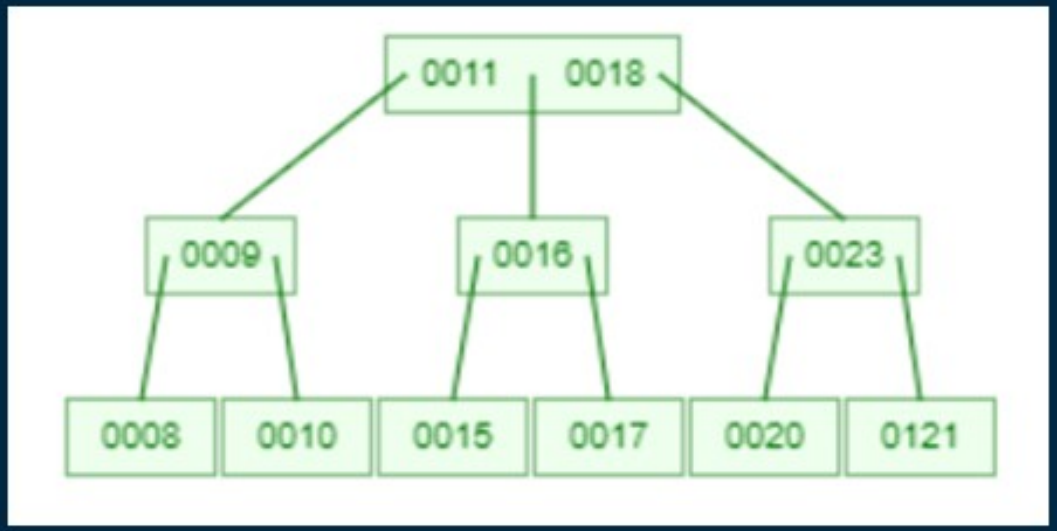
- Adăugare

Exercițiu:
adăugăm valoarea 121

5)



final)



Operații Arbori-B

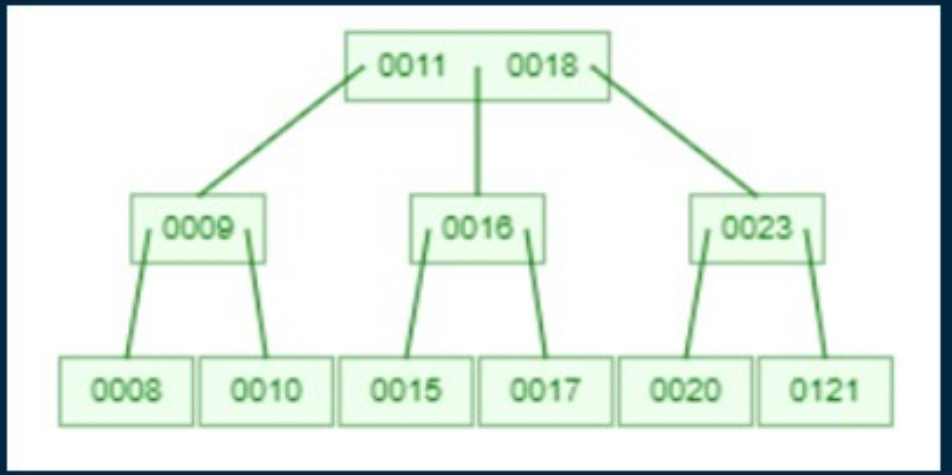
- Ștergere

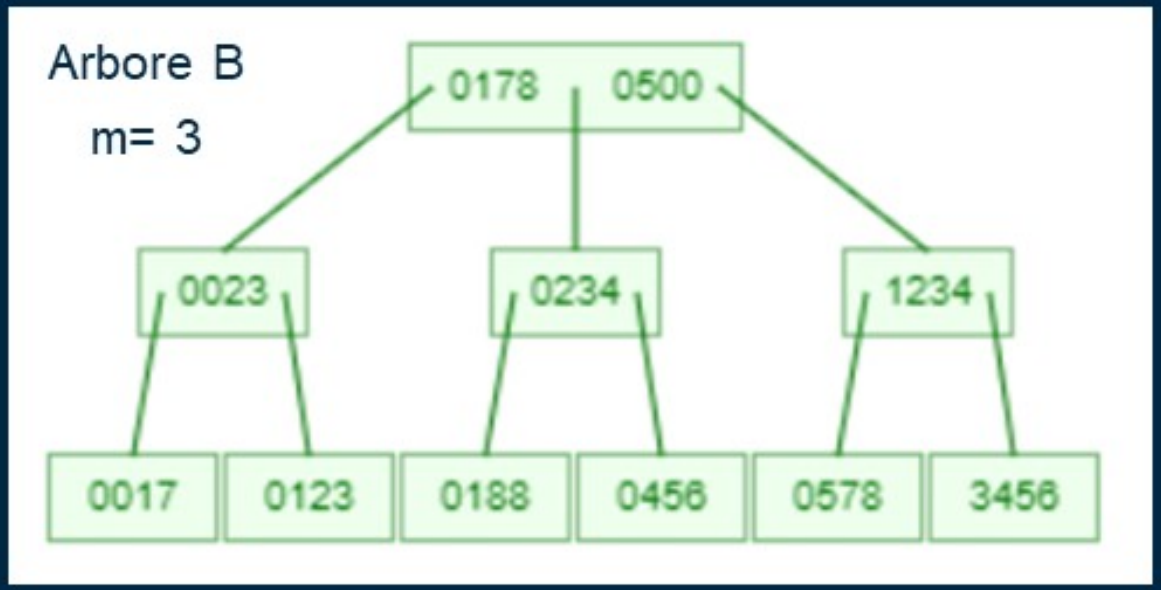
Caz 1: Stergerea unei chei dintr-un nod frunza
a. stergerea nu incalca regulile B-Arbori
b. Stergerea incalca regulile B-Arbori

Caz 2: Stergerea unei chei dintr-un nod interior

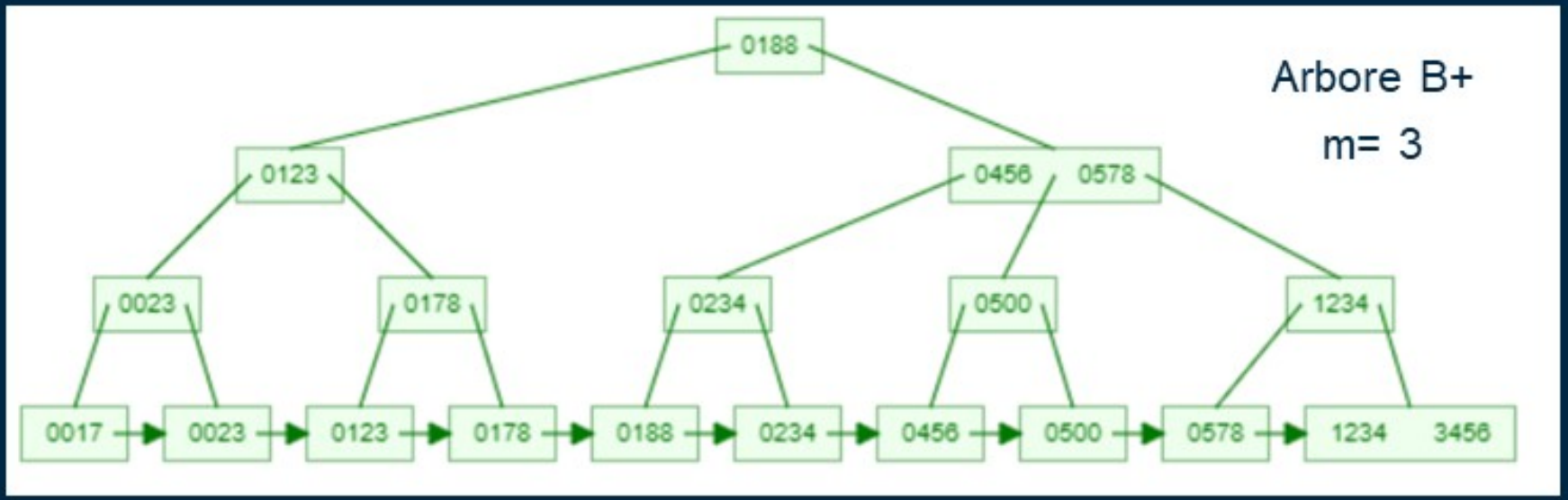
Caz 3: Stergerea unei chei modifica intregul arbore

Arbore-B





17,23,123,178,188,234,456,500,578,1234,3456



Arbori-B – complexitate (timp)

Operația	Caz favorabil	Caz mediu	Caz nefavorabil
Căutare	O(log n)	O(log n)	O(log n)
Adăugare	O(log n)	O(log n)	O(log n)
Ștergere	O(log n)	O(n)	O(n)

Aplicații ale Arbori-B

- Baze de date – indexare date
- Motoare de căutare

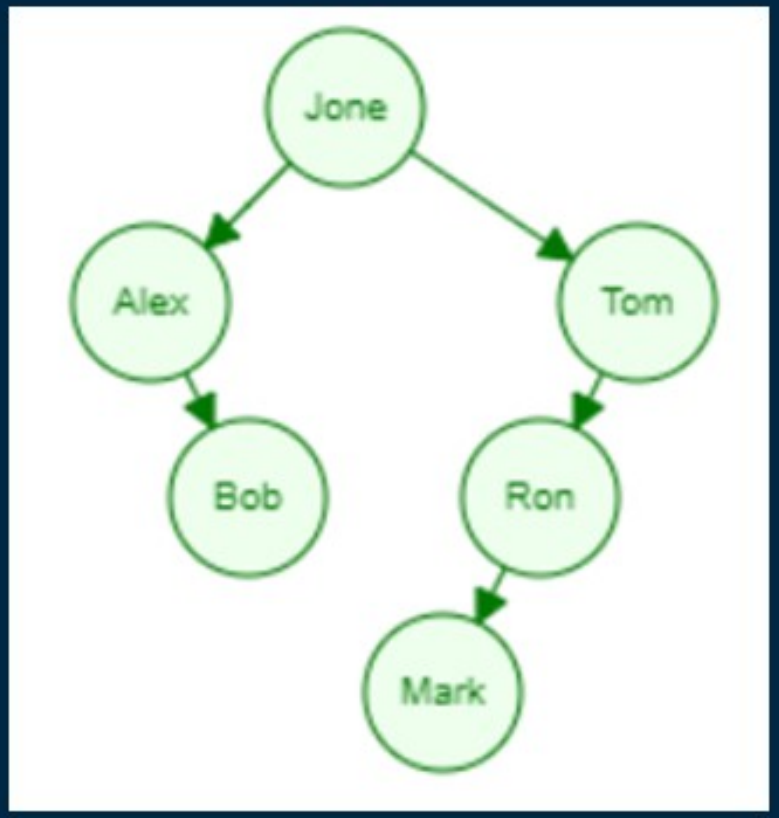
Aplicații ale Arbori-B

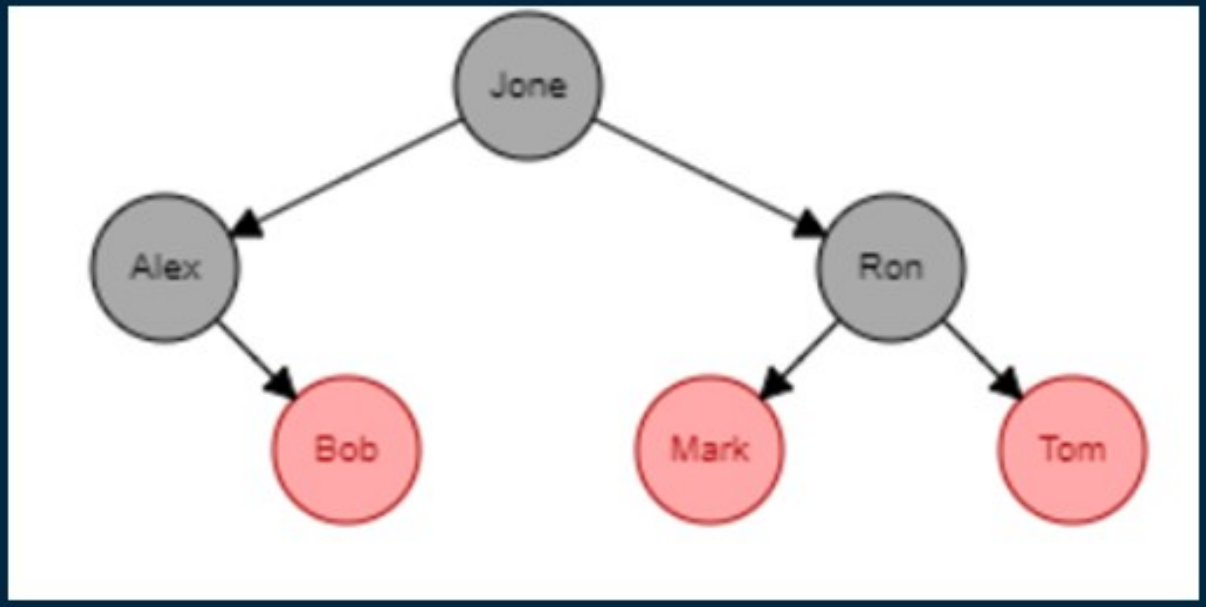
Fie urmatoarea baza de date ca in imaginea de mai jos.

Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32

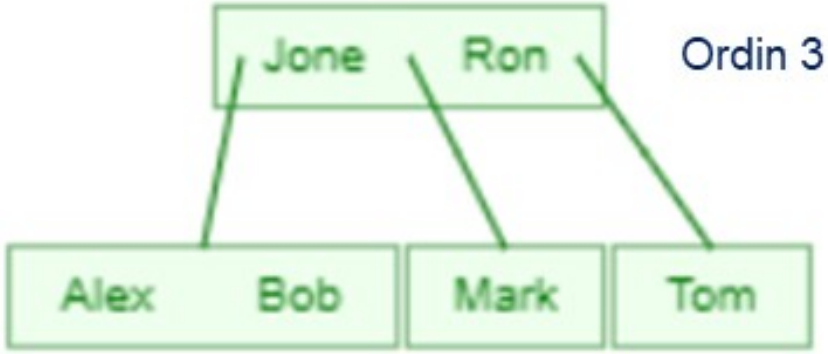
ABC si ARN

Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32

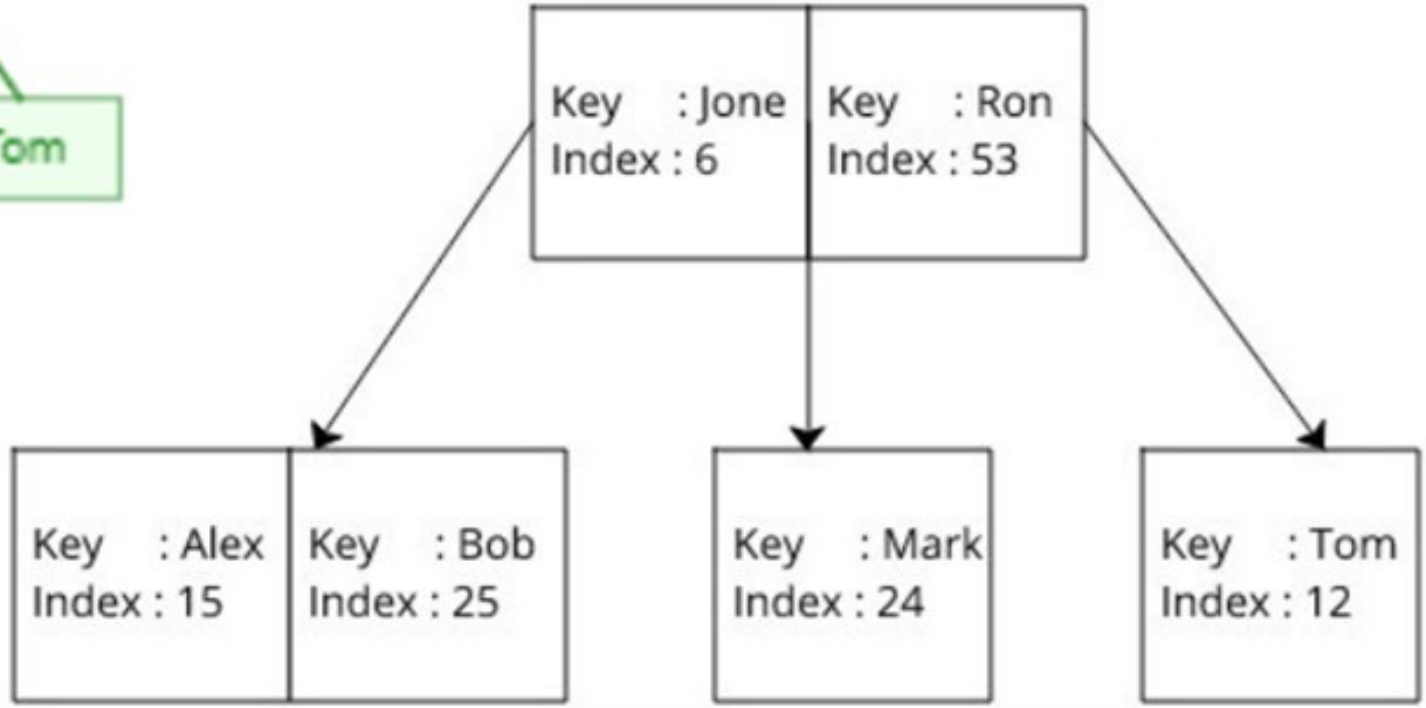




Aplicații ale Arbori-B



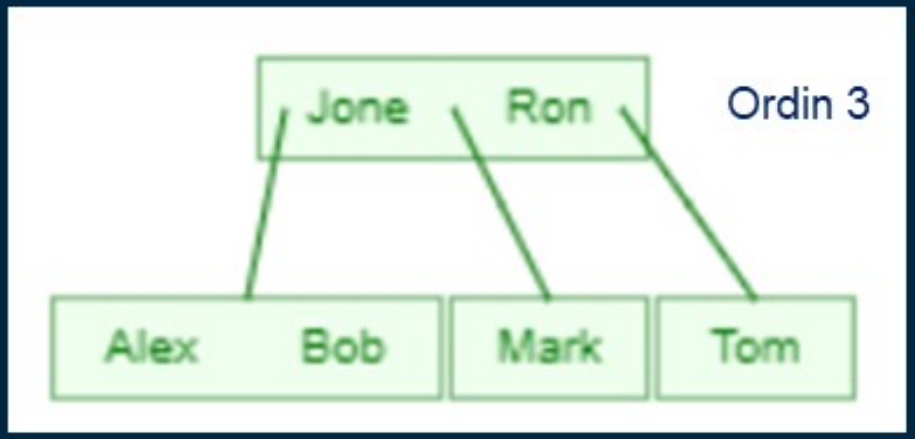
Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32

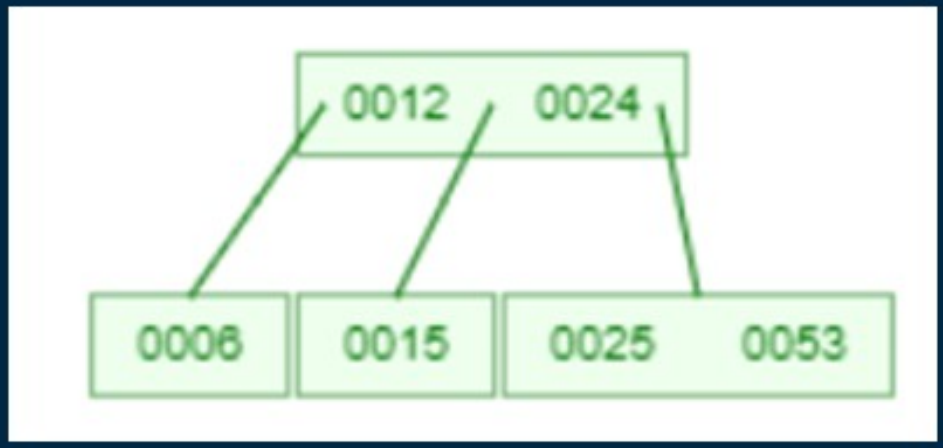


Aplicații ale Arbori-B

Arbore B

Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32

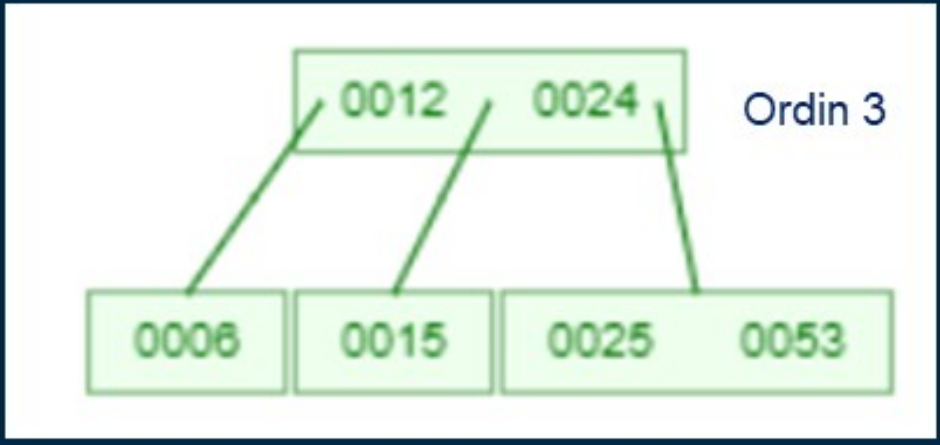




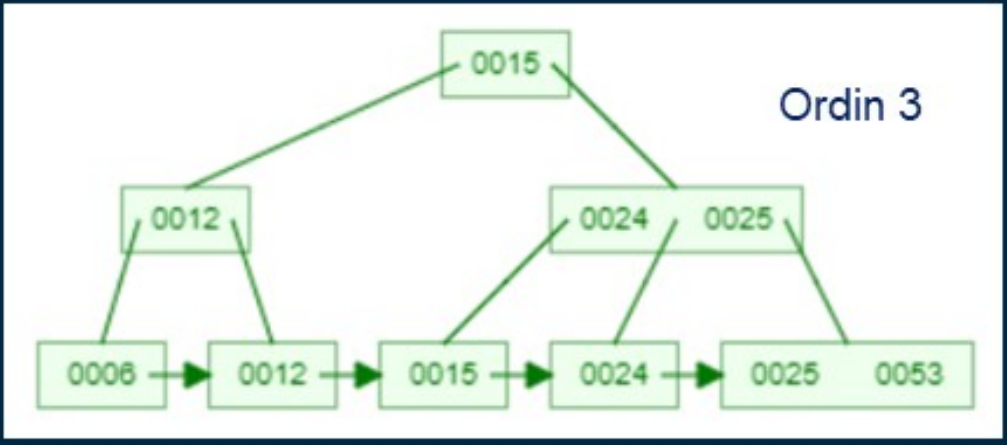
Aplicații ale Arbori-B

Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32

Arbore B

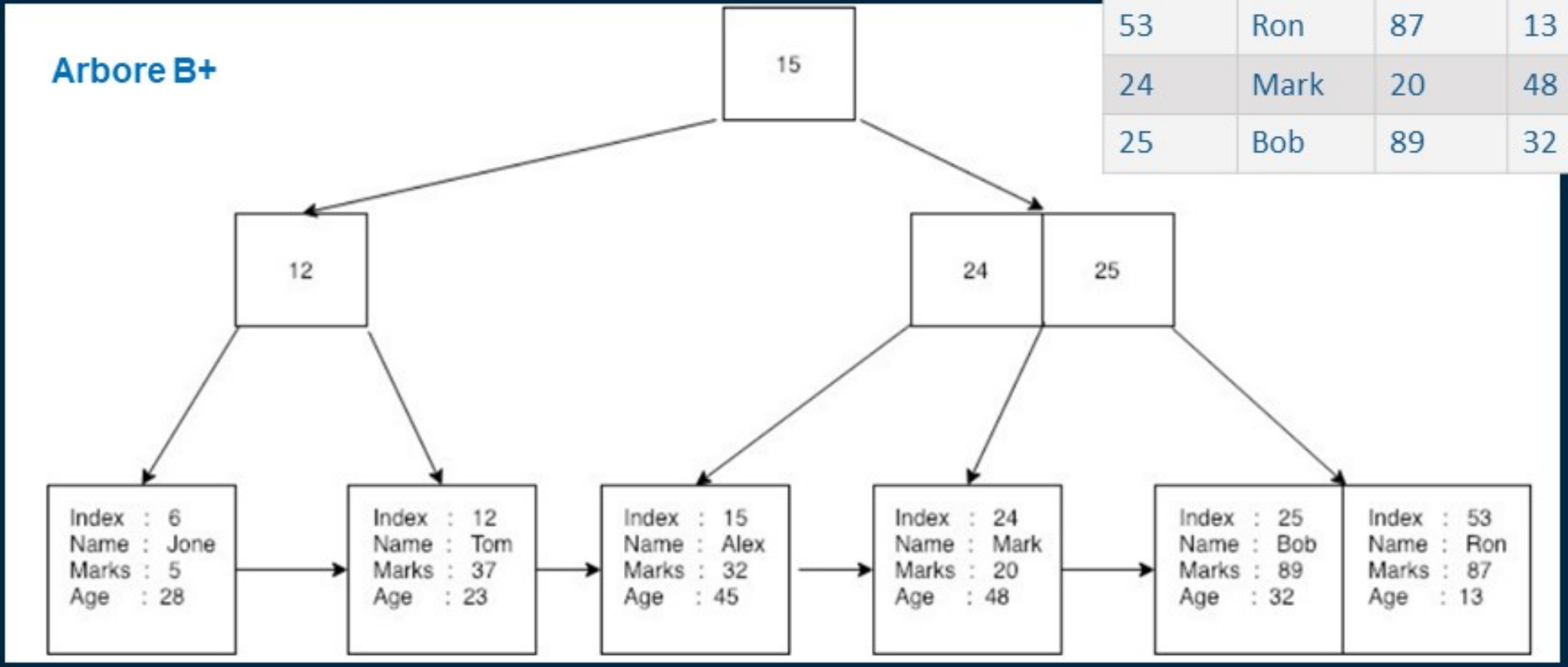


Arbore B+



Aplicații ale Arbori-B

Index	Name	Mark	Age
6	Jone	5	28
15	Alex	32	45
12	Tom	37	23
53	Ron	87	13
24	Mark	20	48
25	Bob	89	32



Intrebari?

dorin.lordache@365.univ-ovidius.ro

Mulțumesc