ADATSZERKEZETEK ÉS ALGORITMUSOK

Python nyelven

Fa adatszerkezetek II.

Kiegyensúlyozott fák

Piros-feketa fa

AVL fa

B-fa adatszerkezet.

Tökéletesen kiegyensúlyozott fák

- Def. Ha minden csúcsnak a bal és a jobb oldali részfájában lévő csúcsok száma legfeljebb eggyel tér el.

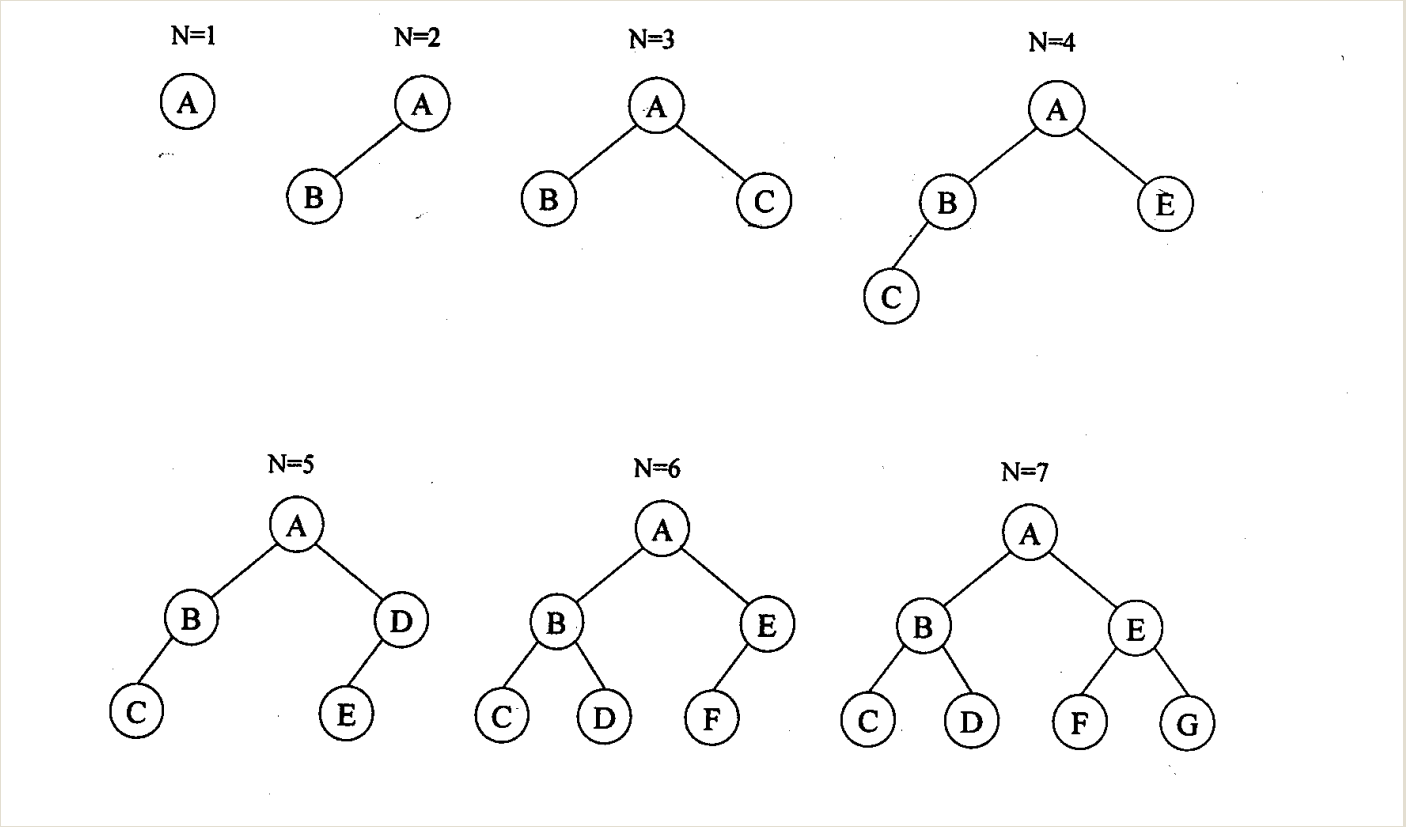
Kiegyensúlyozás

Ha N=0 a fa üres

Ha N>0

1. Használjunk el egy csúcsot gyökérnek
2. Készítsük el az N/2 csúcsból álló bal részfát
3. Készítsük el az N-(N/2)-1 csúcsból álló jobb részfát

Kiegyensúlyozás



Kiegyensúlyozott fák

Def. Ha minden csúcsra legfeljebb egy a különbség a bal és a jobboldali részfa magassága között.

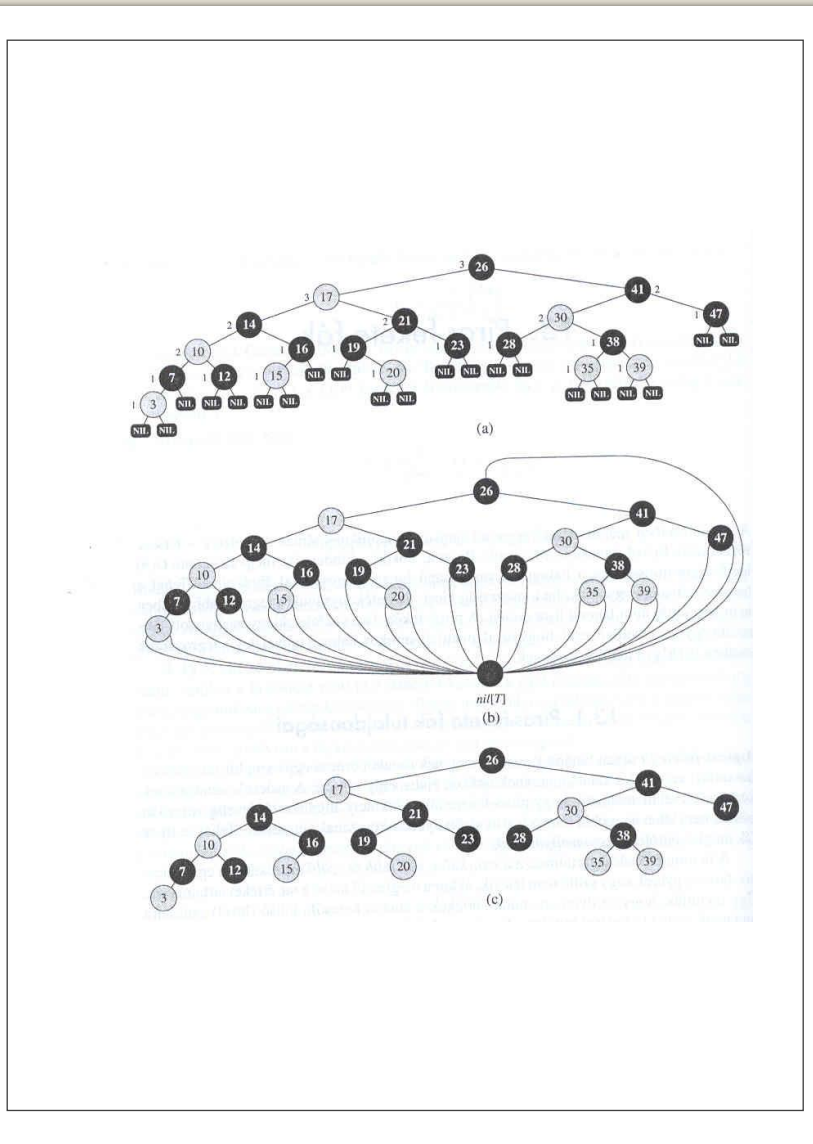
Ha a keresőfában nagy a magasságbeli különbség a szintek között, akkor a keresés időigénye függ a keresendő elem értékétől.

Piros-fekete fák

- Piros-fekete fa olyan bináris keresőfa,melynek minden csúcsa egy extra bit információt hordoz.
- A színek korlátozásával biztosítható, hogy a gyökértől a levélig vezető út hossza nem lehet nagyobb, mint a legrövidebb ilyen út hosszának a kétszerese
- Piros-fekete fák megközelítőleg kiegyensúlyozottak.
- Csúcs adatai: szín, kulcs, bal, jobb, szülő

Tulajdonságok

1. Minden csúcs színe piros vagy fekete
2. Gyökér színe fekete
3. Levél:fekete (gyakran nem is ábrázoljuk)
4. Piros csúcs mindkét gyereke fekete
5. Bármely csúcsból bármely levélig vezető úton ugyanannyi fekete csúcs van.



Piros-fekete fa

- Fekete mélység: Adott csúcsból, ahol ő maga nem számít az eredménybe, a levélig vezető úton a fekete csúcsok száma.

Műveletek

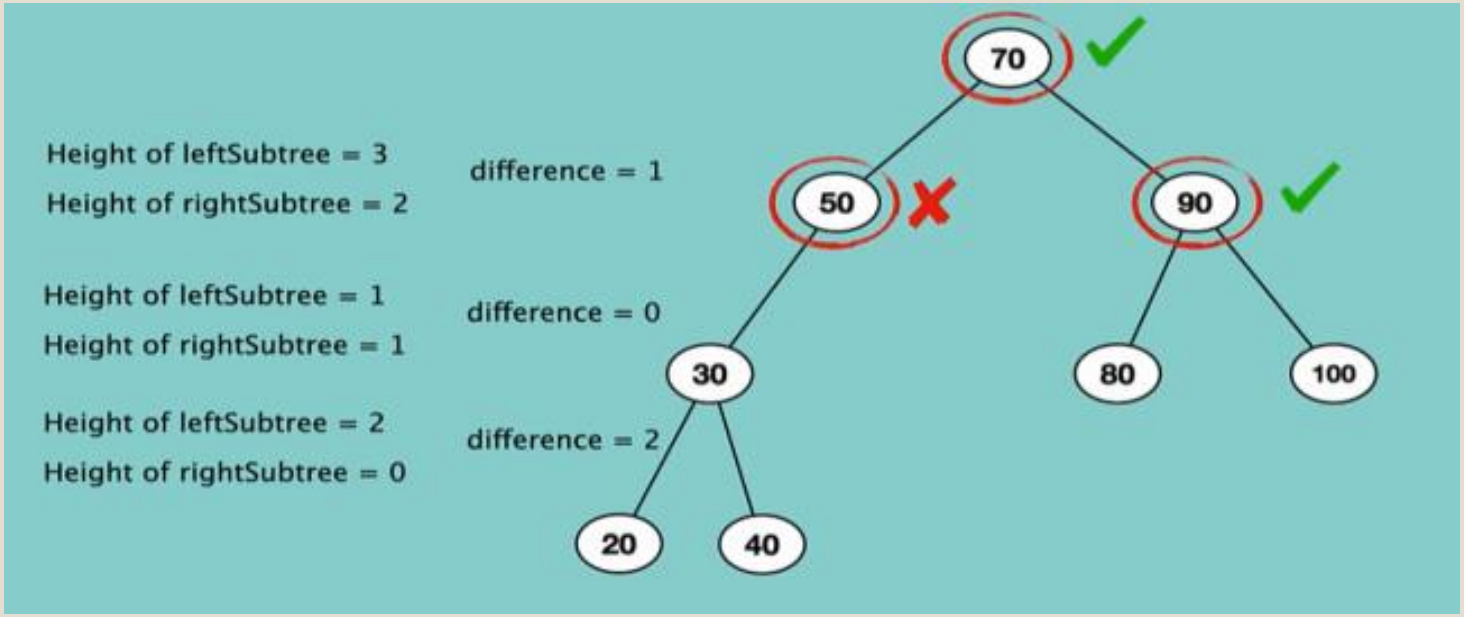
- Forgatások (balra, jobbra forgat)
- Beszúrás
- Törlés

AVL-fa

- Az AVL fa olyan bináris keresőfa amely magasság kiegyensúlyozott:
  - A bal és jobb részfa magassága legfeljebb 1-gyel különbözik.
- Megvalósítás +mező :magasságkülönbség

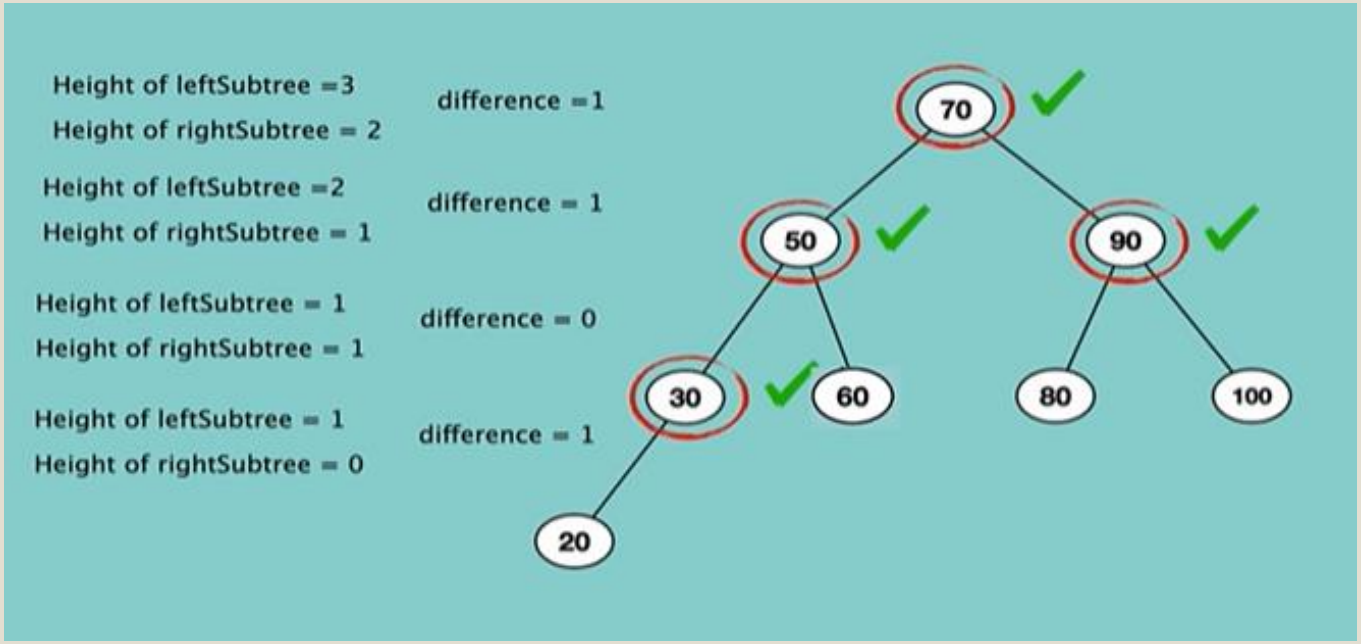
AVL fa

- Magasság-kiegyensúlyozott bináris keresőfa



AVL tulajdonság ellenőrzése

- Magasság-különbség maximum 1 lehet.



Miért előnyös az AVL fa

- Keresés időbonyolultság garantáltan O(logN)
- Kegyenlítés sem vez túl sok műveletet igénybe

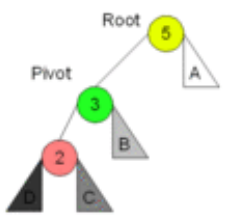
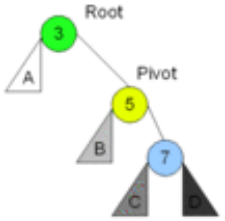
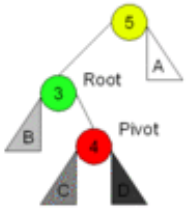
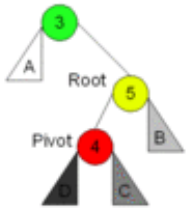
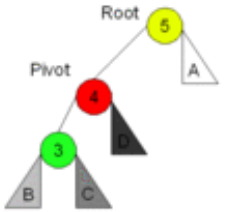
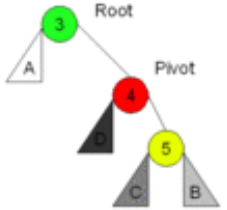
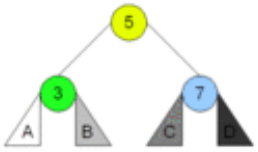
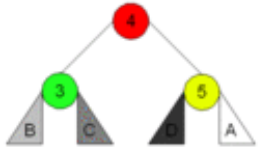
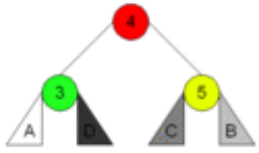
Műveletek AVL fával

- Létrehozás
- Beszúrás
- Keresés
- Bejárás
- Törlés fából
- Fa törlése

AVL fa műveletei - Beszúrás

There are 4 cases in all, choosing which one is made by seeing the direction of the first 2 nodes from the unbalanced node to the newly inserted node and matching them to the top most row.

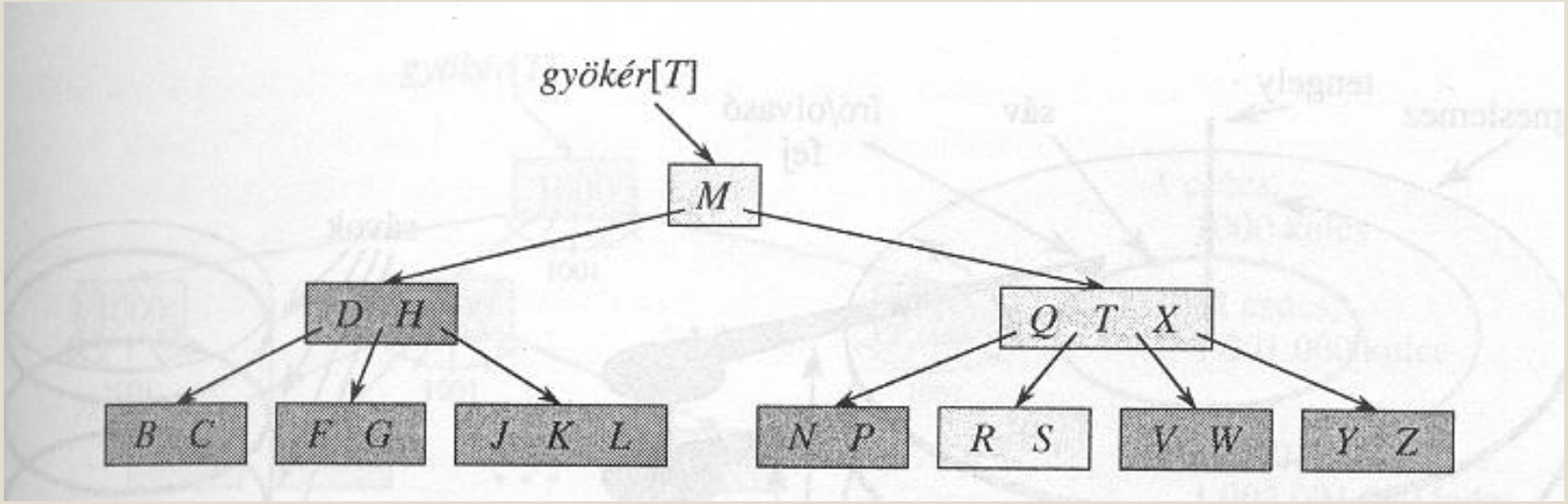
Root is the initial parent before a rotation and **Pivot** is the child to take the root's place.

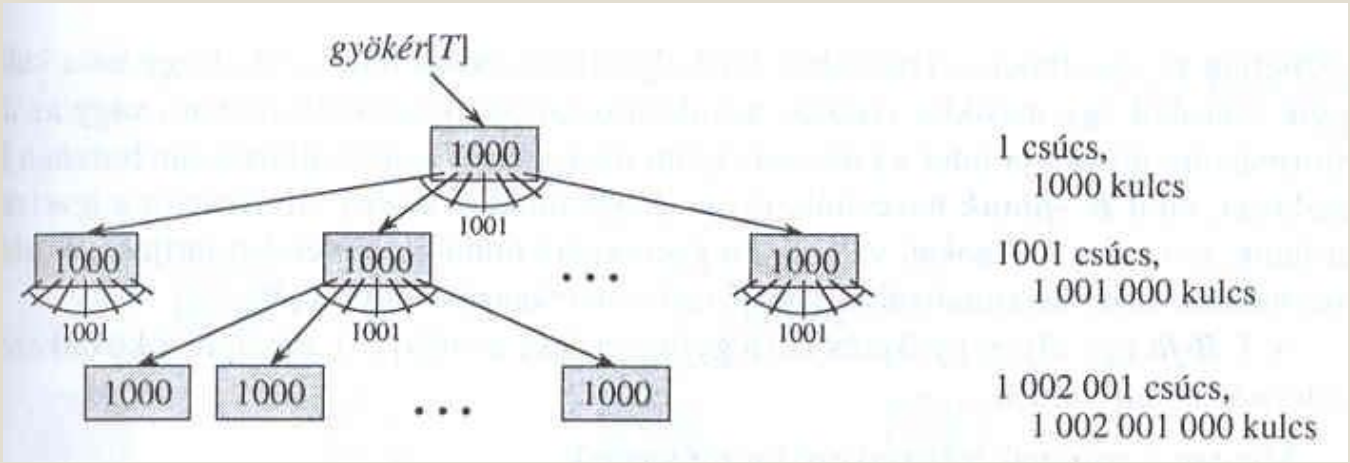
Left Left Case	Right Right Case	Left Right Case	Right Left Case
			
Right Rotation	**Left Rotation**	**Left Rotation**	**Right Rotation**
			
		Right Rotation	**Left Rotation**
			

B-fa

- Nagyméretű keresőfák esetén előfordul, hogy az egész fa nem fér el az operatív tárban.
- A fát részfákra bontjuk, és a részfákat (lapoknak nevezzük) olyan egységként ábrázoljuk, amelyek elemeire együtt hivatkozhatunk.
- Minden lap n és 2n közötti tételt tartalmaz

B-fa definíciója





B-fa tulajdonságai

1. Minden lap legfeljebb 2n tételt tartalmaz
2. Minden lapon a gyökér lapjának kivételével legalább n tétel van.
3. Egy lap lehet levéllap(utód nélküli), vagy m+1 utóddal rendelkezik, ahol m a lapon levő kulcsok száma
4. Minden levéllap ugyanazon a szinten helyezkedik el.

Forrás: http://www.doksi.hu

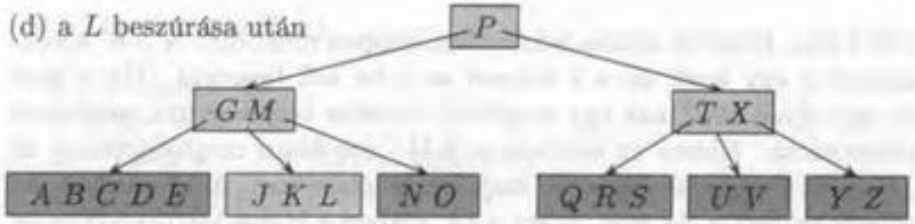
Beszúrás

(a) a kezdeti fa

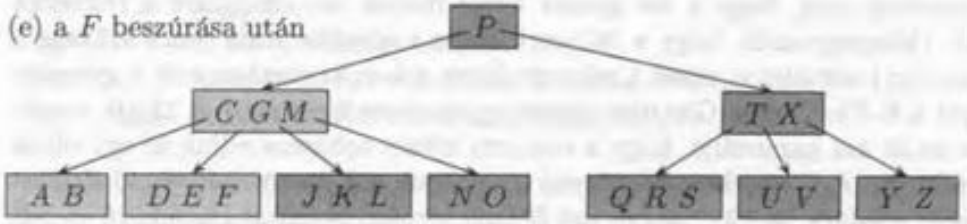
(b) a B beszúrása után

(c) a Q beszúrása után

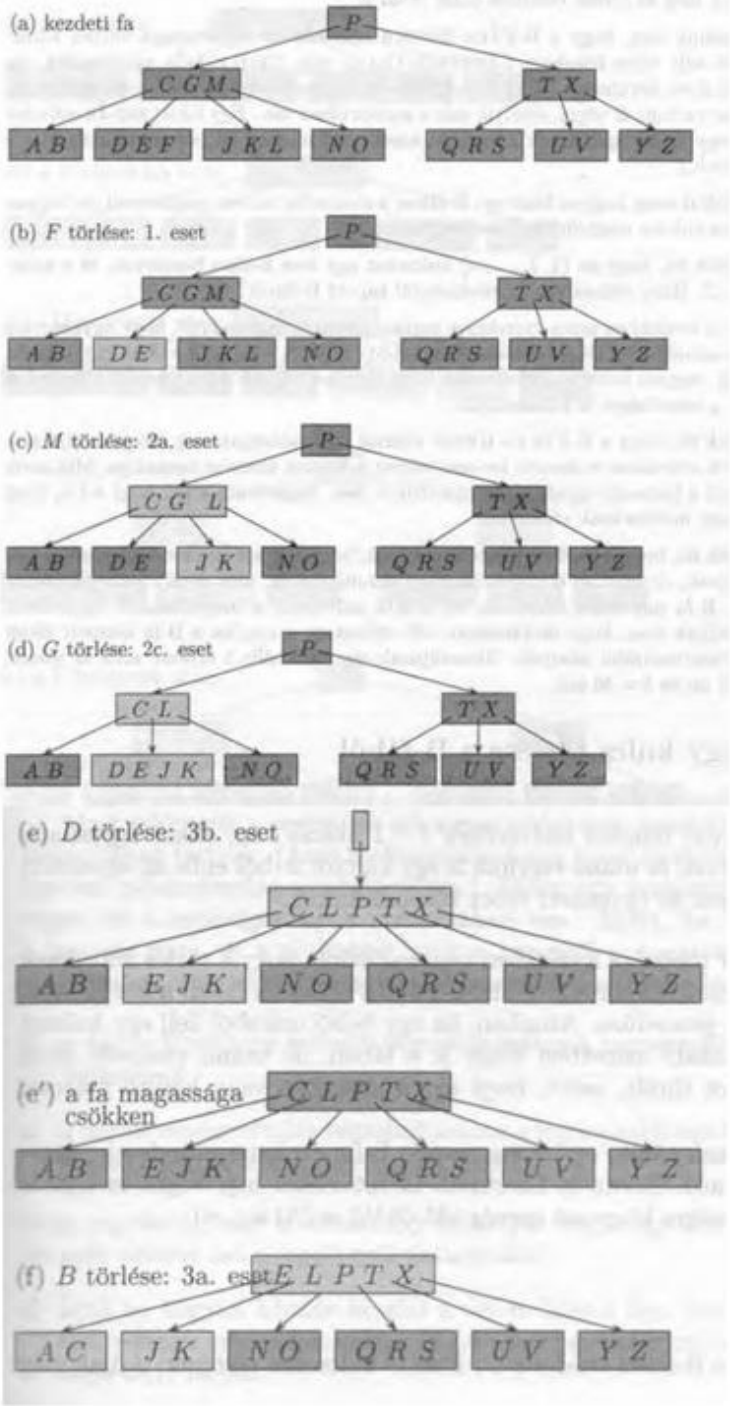
(d) a L beszúrása után



(e) a F beszúrása után



Törlés


(a) kezdeti fa
P
C G M
T X
A B
D E F
J K L
N O
Q R S
U V
Y Z
(b) F törlése: 1. eset
P
C G M
T X
A B
D E
J K L
N O
Q R S
U V
Y Z
(c) M törlése: 2a. eset
P
C G L
T X
A B
D E
J K
N O
Q R S
U V
Y Z
(d) G törlése: 2c. eset
P
C L
T X
A B
D E J K
N O
Q R S
U V
Y Z
(e) D törlése: 3b. eset
C L P T X
A B
E J K
N O
Q R S
U V
Y Z
(e') a fa magassága csökken
C L P T X
A B
E J K
N O
Q R S
U V
Y Z
(f) B törlése: 3a. eset
E L P T X
A C
J K
N O
Q R S
U V
Y Z

Köszönöm a figyelmet!