

# Struktura indeksa: B-stablo

<http://cis.stvincent.edu/html/tutorials/swd/btree/btree.html>

# Uvod

- ISAM (Index-Sequential Access Method, IBM sredina 60-tih godina 20. veka)
- Nedostaci: sekvencijalno pretraživanje indeksa
- Poboljšanje: indeks nad indeksom
- Uopštavanje: drvoidna indeksna struktura
  - svaki nivo osim nivoa lista je redak indeksa
  - ilustracija

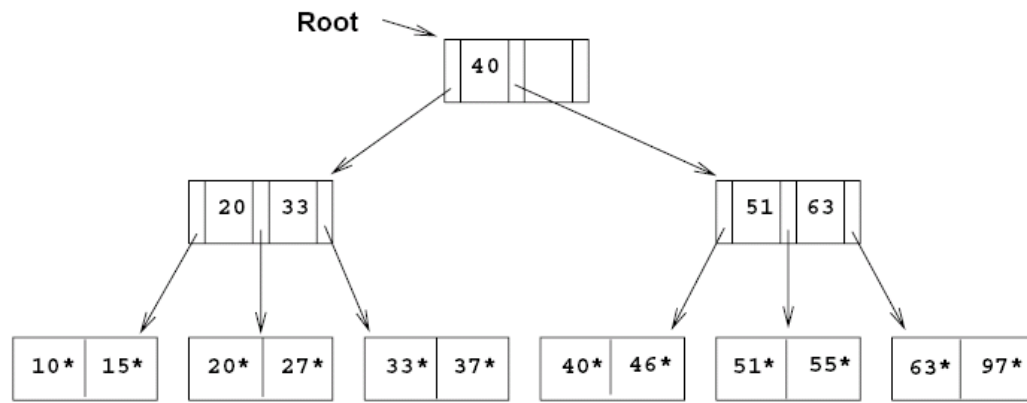


Figure 9.5 Sample ISAM Tree

# ISAM

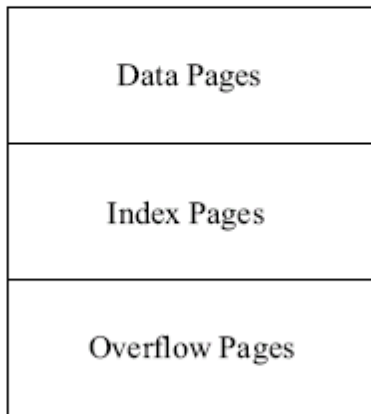


Figure 9.4 Page Allocation in ISAM

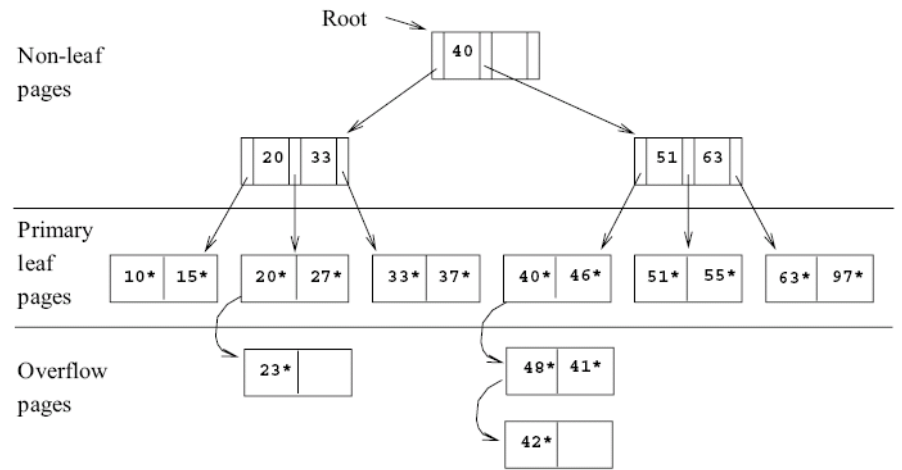


Figure 9.6 ISAM Tree after Inserts

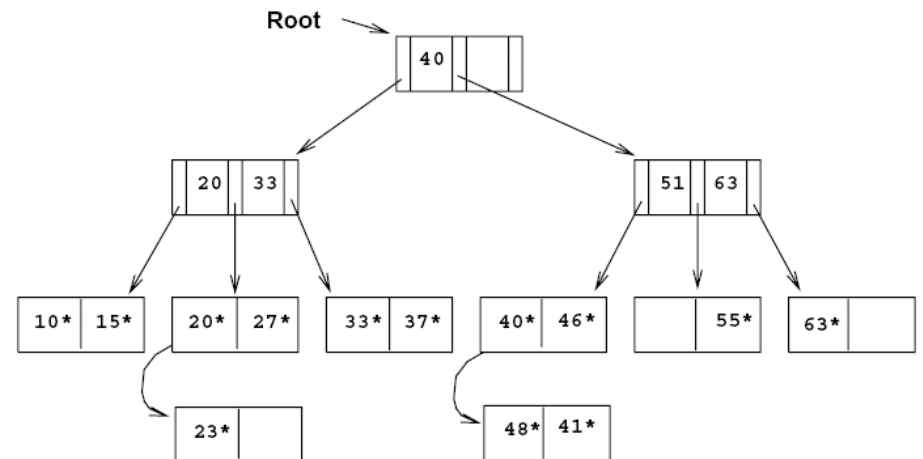


Figure 9.7 ISAM Tree after Deletes

# Definicije

## Graf

- *Usmereni graf*  $\Gamma$  je uređeni par  $(A, r)$ , gde je  $A$  – konačni skup a  $r$  – binarna relacija nad skupom  $A$ .
- Elemente skupa  $A$  zovemo *čvorovi*, a elemente skupa  $r$  – *potezi*. Ako je  $(a_i, a_j) \in r$  poteg, kaže se da je čvor  $a_i$  – *neposredni prethodnik* čvora  $a_j$ , a da je čvor  $a_j$  – *neposredni sledbenik* čvora  $a_i$ ; takođe, čvor  $a_i$  je *izlazni čvor* potega  $(a_i, a_j)$ , a čvor  $a_j$  je *ulazni čvor* potega  $(a_i, a_j)$ .
- *Ulazni red* čvora  $a_i$  je broj potega kojima je taj čvor – ulazni čvor, a *izlazni red* čvora  $a_i$  je broj potega kojima je taj čvor – izlazni čvor.
- *Putanja dužine*  $n$  od čvora  $a_i$  do čvora  $a_k$  u grafu  $\Gamma$  je niz čvorova  $(a_i, a_{i+1}, \dots, a_{i+n})$ , takvih da je  $a_{i+n} = a_k$  i  $(a_j, a_{j+1}) \in r$  za svako  $j = i, i + 1, \dots, i+n-1$ . Putanja od čvora  $a_i$  do čvora  $a_k$ , dužine veće od 1, za koju važi da je  $a_i = a_k$ , zove se *ciklus*.

# Stablo

## Stablo

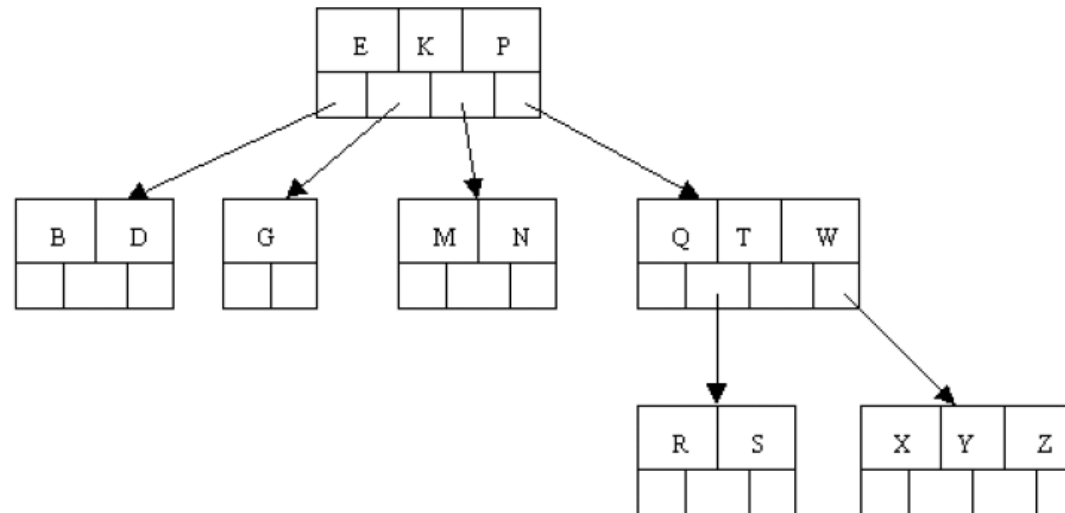
Graf  $\Gamma$  je *stablo* ako zadovoljava sledeće uslove:

- ulazni red svakog čvora (osim jednog) je 1;
  - ulazni red jednog čvora je 0, i taj čvor se zove *koren* stabla;
  - izlazni red svakog čvora je  $\geq 0$  i konačan;
  - graf  $\Gamma$  ne sadrži cikluse.
- 
- čvorovi bez neposrednih sledbenika zovu se *listovi* stabla (izlazni red svakog lista je 0);
  - *visina* stabla je maksimalna dužina putanje od korena do nekog lista stabla.
  - čvorovi – neposredni sledbenici istog čvora zovu se *braća*

# Stablo reda $m$

- Stablo reda  $m$  je stablo u kome je izlazni red svakog čvora manji ili jednak  $m$  (svaki čvor ima najviše  $m$  sledbenika).
- Svaki čvor sadrži određen broj *ključeva*; čvor sa  $k$  sledbenika sadrži tačno  $k - 1$  ključeva.
- Ako su ključevi i podstabla uređeni kao kod stabla pretrage, stablo reda  $m$  naziva se **stablom pretrage reda  $m$** .
- Primer: stablo pretrage reda 4 (prvi red čvora prikazuje ključeve, drugi red pokazivače na čvorove sledbenike).
- U realnoj situaciji, umesto ključa biće ceo slog, pa će u prvom redu biti niz slogova koji uključuje ključ i pridružene podatke (ili ključ i redni broj sloga, ako su slogovi smešteni u posebnoj datoteci).

# Stablo pretrage reda m: primer



# Stablo pretrage reda m: osobine

- Čvor stabla pretrage reda m može da se predstavi strukturom sa brojem ključeva (*int Broj*) i dva niza: nizom *Ključevi* od m-1 ključeva nekog tipa i nizom *Grane* od m pokazivača na strukture istog tipa
- Struct NodeType  
{  
    int Broj; // broj ključeva smeštenih u tekućem čvoru  
    tipKljuča Ključevi[m-1]; // niz za najviše m-1 ključeva  
    struct NodeType \*Grane [m]; // niz pokazivača  
} N;
- U tom slučaju, stablo pretrage reda 4 (slično za m) treba da zadovolji sledeće uslove u vezi sa uređenjem ključeva:



# Stablo pretrage reda m: osobine

- Ključevi u svakom čvoru su u rastućem poretku.
- U svakom čvoru N važi:
  - Podstablo koje počinje u strukturi N.Grane[0] sadrži ključeve koji su manji od N.Ključevi[0].
  - Podstablo koje počinje u strukturi N.Grane[1] sadrži ključeve koji su veći od N.Ključevi[0] i istovremeno manji od N.Ključevi[1].
  - Podstablo koje počinje u strukturi N.Grane[2] sadrži ključeve koji su veći od N.Ključevi[1] i istovremeno manji od N.Ključevi[2].
  - Podstablo koje počinje u strukturi N.Grane[3] sadrži ključeve koji su veći od N.Ključevi[2].
- Ne govorimo o duplikatima

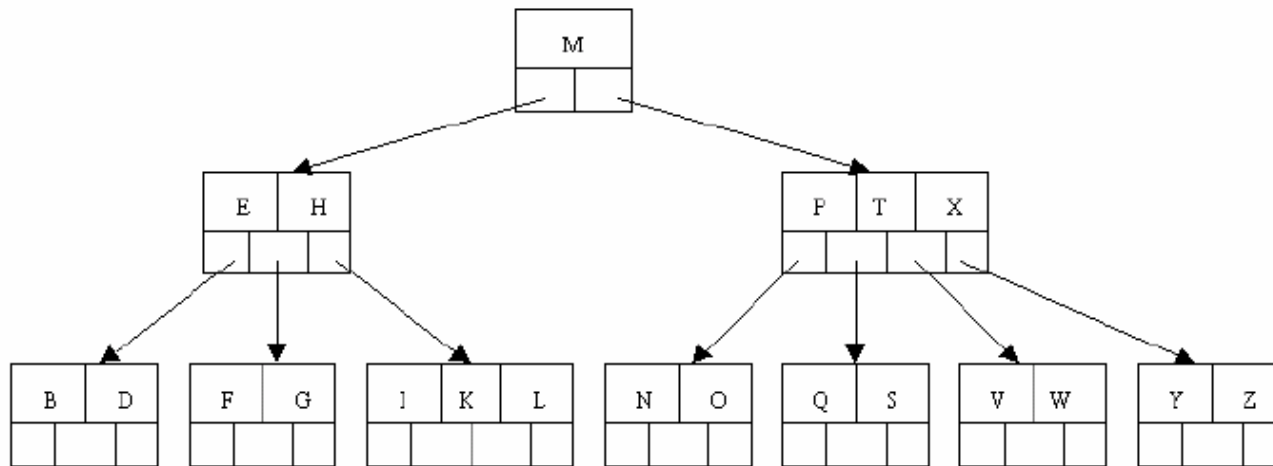
# B-stablo

- Posebno poželjna osobina stabla pretrage reda  $m$ , kada je u pitanju efikasnost njegovog pretraživanja, jeste *balansiranost* koja označava da je dužina puta od korena do svakog lista jednaka.
- Visina stabla:  $\log_m Br$ ,  $Br$  – ukupan broj ključeva u stablu; obično 3-4 (npr. 10000000 ključeva,  $m=50$ , visina 4)
- Jedna specifična struktura balansirano stabla, poznata kao **B-stablo**, poslužila je kao osnov za izgradnju niza struktura podataka koje se koriste u implementaciji indeksa.
- Struktura **specijalizovana za korišćenje na disku**
- **Čvorovi** – blokovi fiksne veličine na disku – “**stranice**”
- Puno ključeva na jednoj stranici, puno sledbenika, veliki red  $m$  – velika efikasnost pristupa podacima na disku

# B-stablo

- B-stablo reda  $m$  je stablo pretrage reda  $m$  takvo da:
  - Svi listovi su na najnižem nivou (stablo je balansirano)
  - Svi unutrašnji čvorovi (osim možda korena) imaju najmanje  $\text{ceil}(m / 2)$  (nepraznih) sledbenika.
  - Koreni čvor, ako nije list, ima najmanje 2 sledbenika, a ako je istovremeno i list onda nema sledbenika (i stablo se sastoji samo od tog jednog čvora)
  - Svaki list sadrži najmanje  $\text{ceil}(m / 2) - 1$  ključeva
- B-stablo je osnovna drvoidna indeksna struktura
- Sadrži podatke (ključeve tj. slogove) **na svim nivoima**
- Popunjenost stranice min 50%, u proseku oko 70%
- Modifikacije: B\* stablo, B+ stablo, sa većom popunjenošću / razdvojenim indeksom od podataka / sekvencijalnošću nad podacima

# B-stablo: primer - red 5



# Operacije nad B-stablom: pretraživanje

- (1) Neka je stranica  $N$  - korena stranica
- (2) Pretražiti stranicu  $N$  na ključ  $Klj$
- (3) Ako je ključ  $Klj$  nađen, pretraživanje uspešno
- (4) Inače,
  - Ako je stranica  $N$  list, pretraživanje je neuspešno
  - Inače, naći među ključevima  $N.Kljucevi[0]$ ,  $N.Kljucevi[1]$ ,  $N.Kljucevi[2]$ , ... ,  $N.Kljucevi[Br-1]$  najmanji ključ koji je veći od  $Klj$ ; neka je to ključ  $N.Kljucevi[j]$ ;  $N$  postaje stranica na koju pokazuje pokazivač  $N.Grane[j-1]$ ; preći na korak 2.
- Primer: pretražiti prethodno stablo na ključeve  $S$ ,  $J$ ,  $A$

# Operacije nad B-stablom: unošenje

- Pretražiti B-stablo ključem unošenja
- Ako ključ nije nađen, pretraga je završena u listu
  - (1) Ako ima mesta u listu, uneti ključ u list uz eventualna pomeranja
  - (2) Ako je list pun, treba ga “pocepati”:
    - (2.1) oko polovine ključeva ostaje na starom listu
    - (2.2) oko polovine ključeva ide na novi list desno od postojećeg
    - (2.3) srednji ključ se “penje” u roditeljski čvor, pomeraju se ključevi i pokazivači

# Operacije nad B-stablom: unošenje (nast.)

- (3) Ako u roditeljskom (unutrašnjem) čvoru nema mesta, i on se “cepa”:
  - (3.1) oko polovine ključeva ostaje na starom čvoru
  - (3.2) oko polovine ključeva ide u novi čvor desno od postojećeg
  - (3.3.) srednji ključ se “penje” u roditeljski čvor, uz pomeranje ključeva i pokazivača (rekurzivno korak 3)
- (4) Ako se “pocepa” koreni čvor, srednji ključ ide u novi koreni čvor; visina stabla se povećava za 1
- Ako je ključ nađen – unošenje se ne vrši (slučaj bez duplikata)

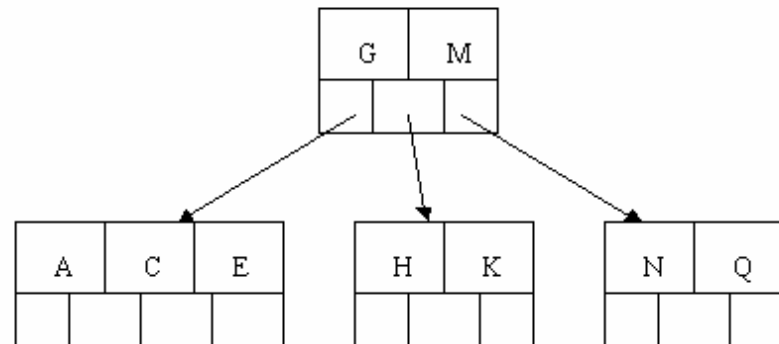
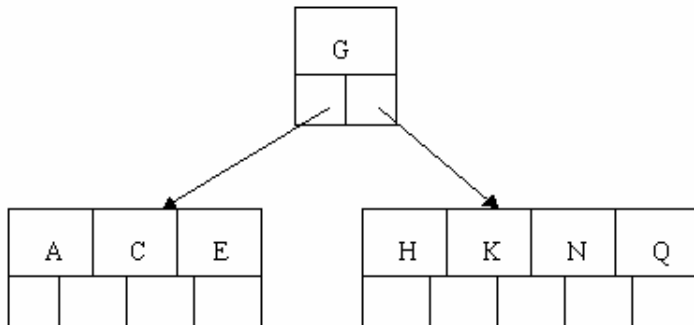
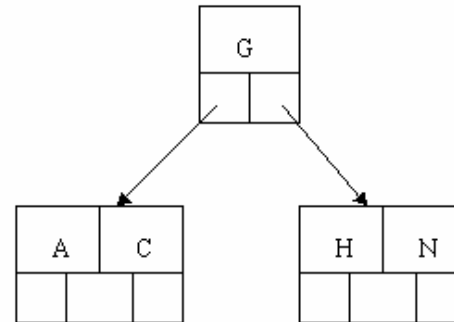
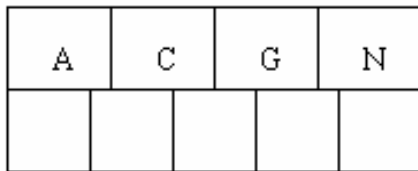
# B-stablo: unošenje - primer

- Unose se sledeća slova u (na početku) prazno B stablo reda 5:
- C N G A H E K Q M F W L T Z D P R X Y S
- “Red 5” znači da čvor može da ima najviše 5 neposrednih sledbenika i 4 ključa (najmanje 3 neposredna sledbenika i 2 ključa).
- Svi čvorovi osim korena moraju da imaju najmanje 2 ključa (koren ima najmanje 1 ključ).
- Prva 4 slova (C N G A ) se unose u isti (prvi) čvor:

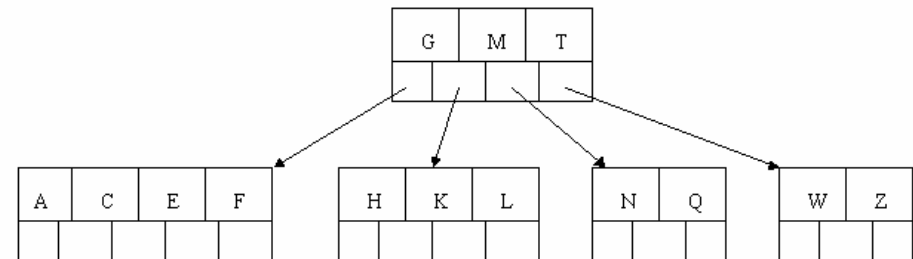
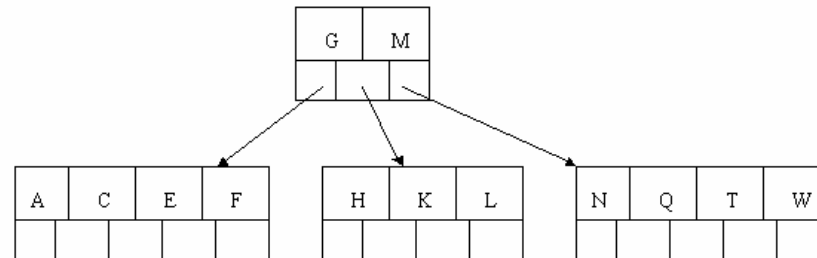
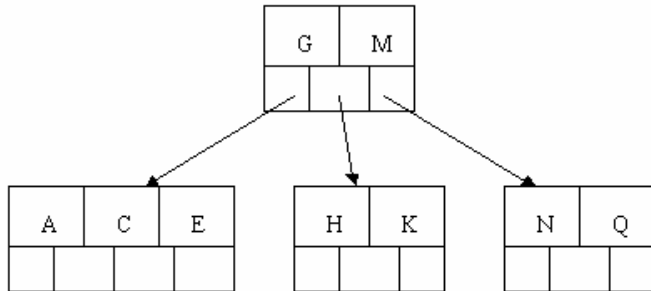
A	C	G	N



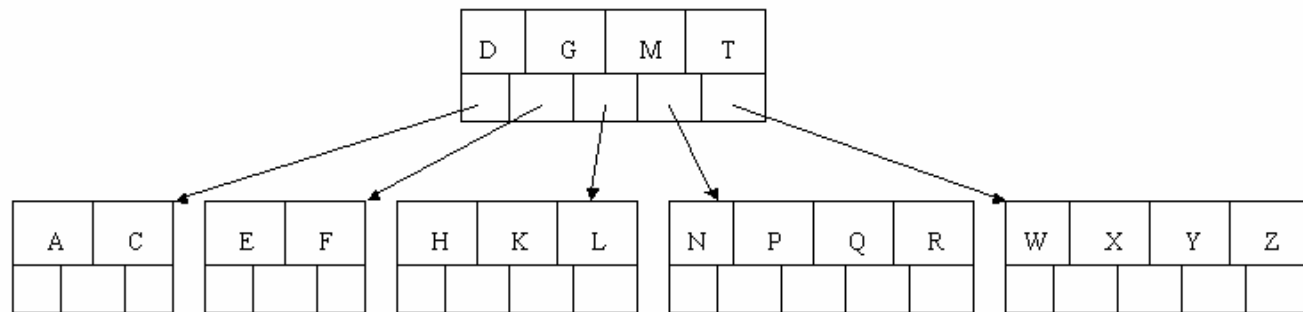
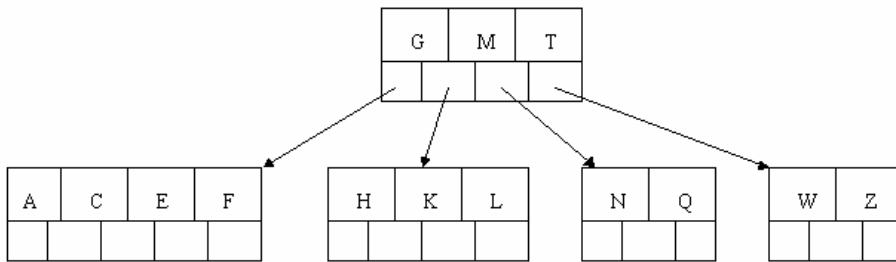
# Unošenje ključeva H E K Q M: cepanje lista (H, 2), (M, 4)



# Unošenje ključeva F W L T Z: cepanje lista (Z, 2)

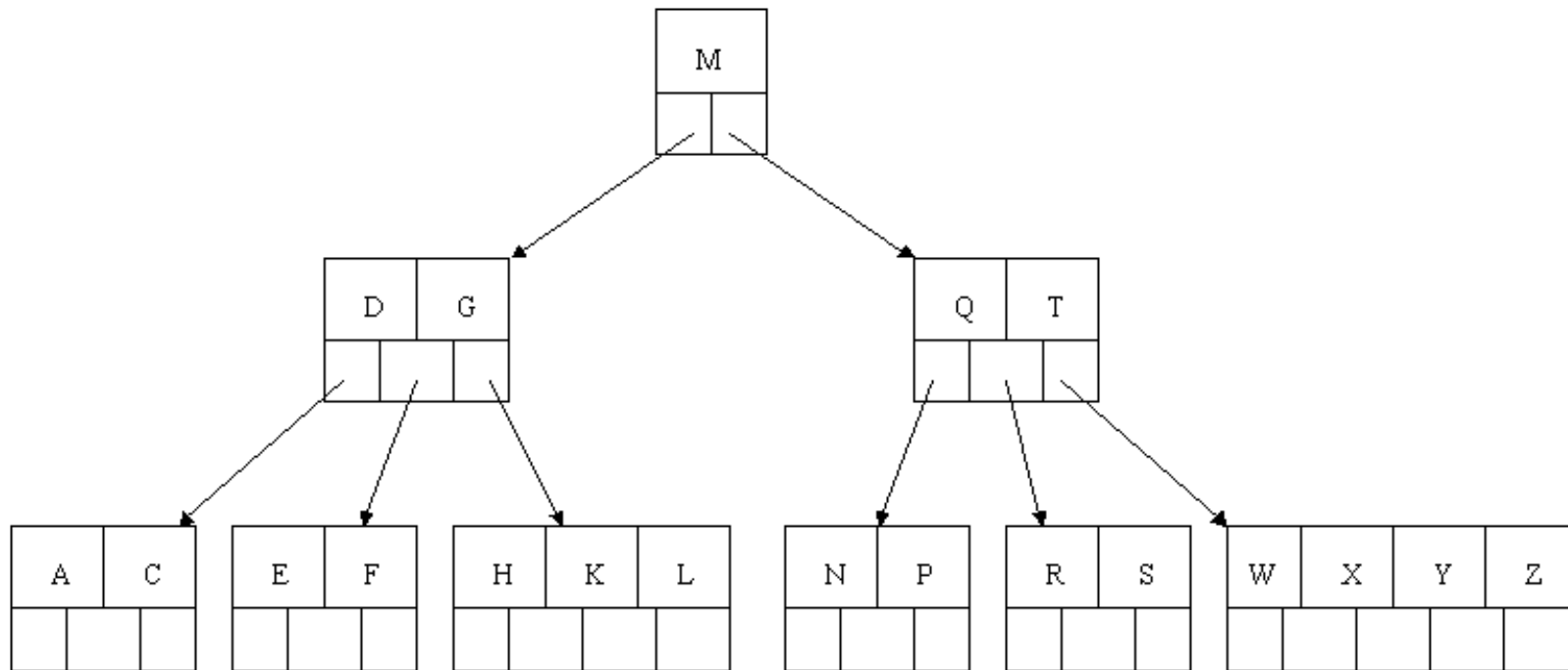


# Unošenje ključeva D P R X Y: cepanje lista (D)



# Unošenje ključa S:

cepanje lista i unutrašnje stranice;  
novi koren, povećanje visine stabla

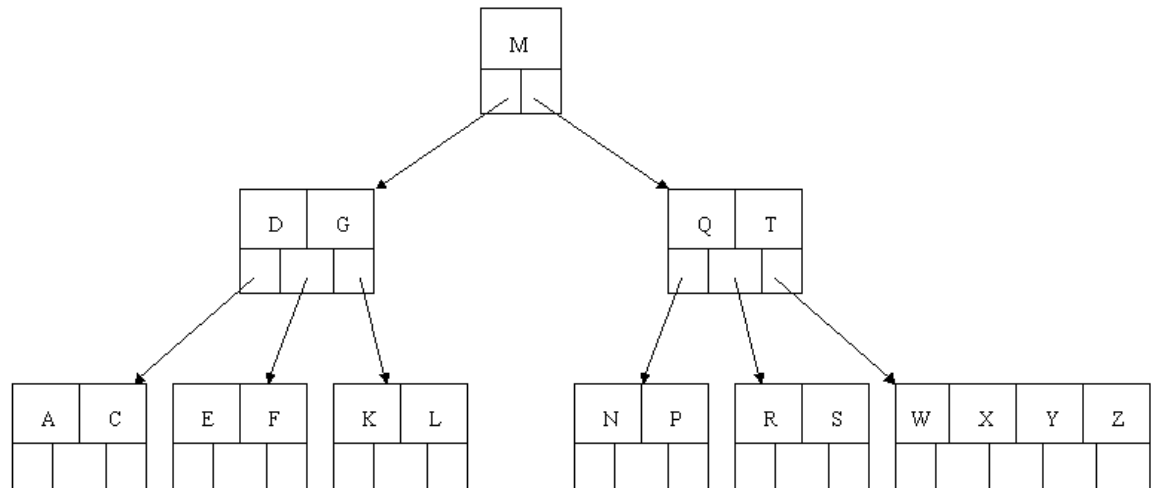
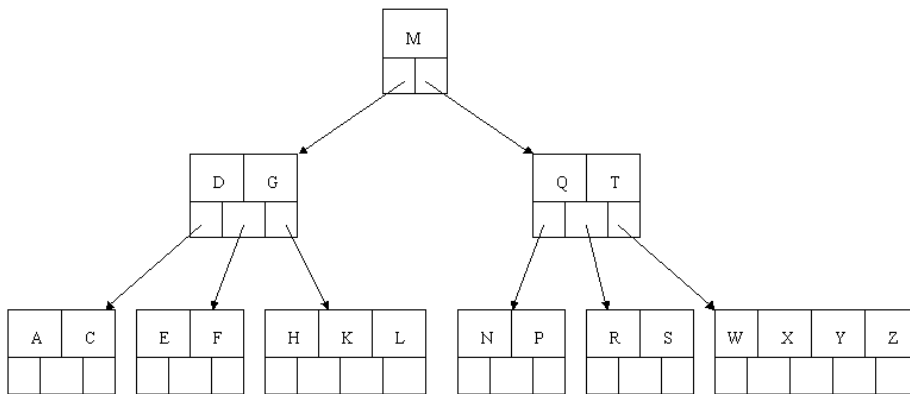


# B-stablo: brisanje - algoritam

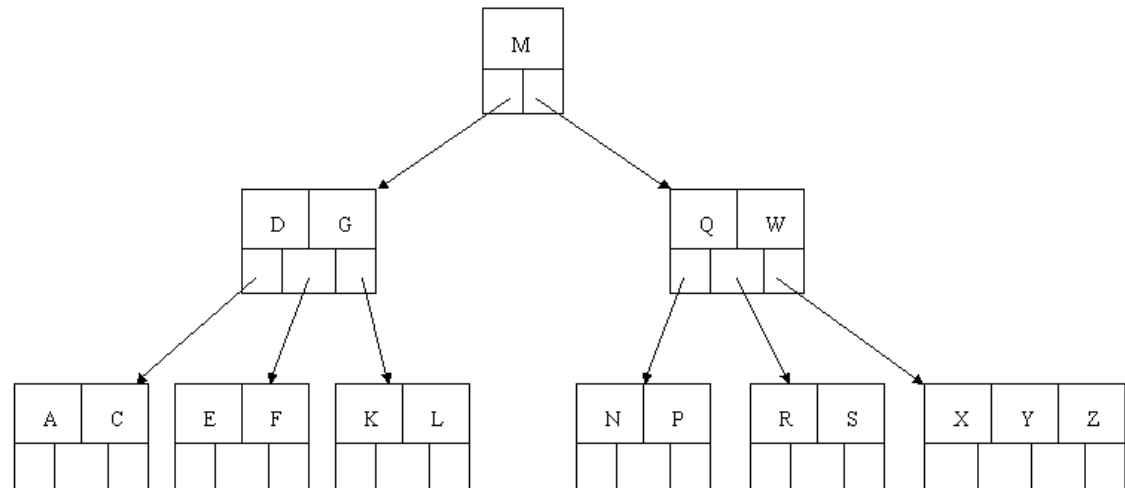
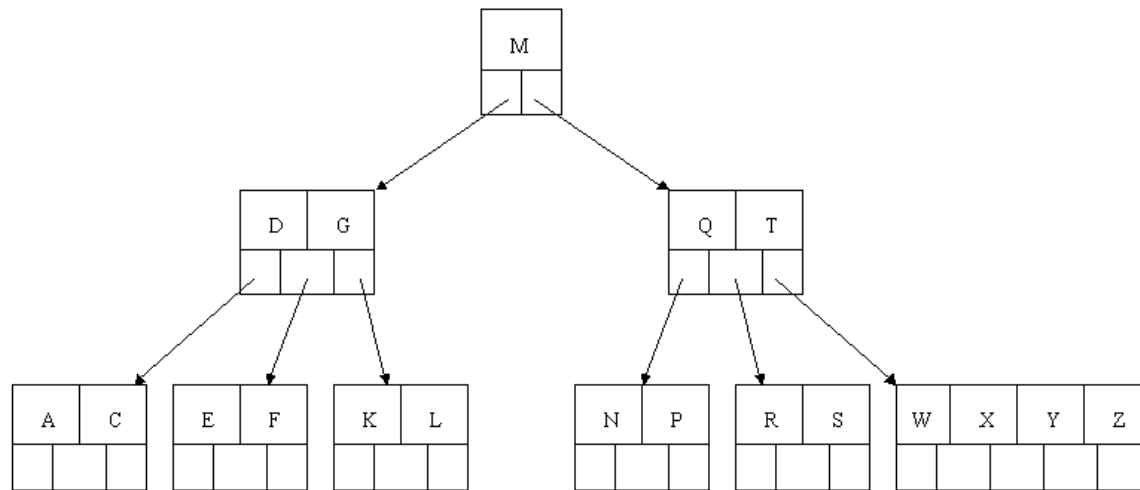
- Brisanje se uvek svodi na brisanje iz lista
- (1) Ako se briše ključ iz lista onda
  - (1.1) Ako ostaje dovoljno ključeva, vrši se samo pomeranje
  - (1.2) Ako ne ostaje dovoljno ključeva onda
    - (1.2.1) Ako susedni brat (čvor) ima “višak” ključeva, najmanji ključ iz desnog (odnosno najveći iz levog brata) se “penje” u čvor-prethodnik, a ključ iz prethodnika se “spušta” u čvor iz kojeg smo izbrisali ključ
    - (1.2.2) Ako nijedan susedni brat nema “višak” ključeva, čvor iz kojeg smo izbrisali ključ spaja se sa levim ili desnim bratom, uz dodavanje “razdvojnog” ključa iz čvora-prethodnika. Ako čvor-prethodnik nema dovoljno ključeva, vrši se rotacija preko njegovog čvora-prethodnika, ili spajanje sa bratom
- (2) Ako se briše ključ iz unutrašnjeg čvora, onda se umesto njega “penje” najmanji ključ iz njegovog sledbenika; ključ koji se penje briše se iz čvora u kome je bio – pravilo se primenjuje rekurzivno i svodi na brisanje iz lista

# B stablo: brisanje - primer

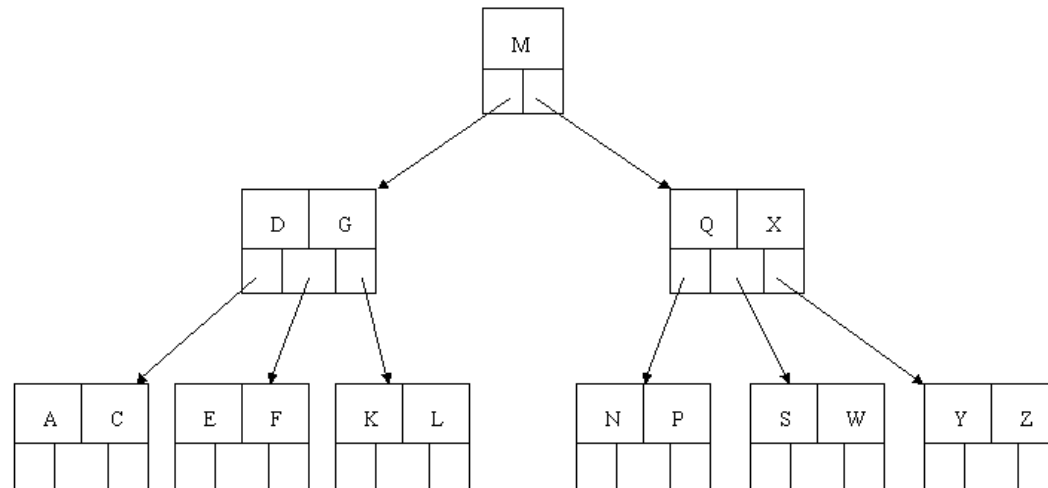
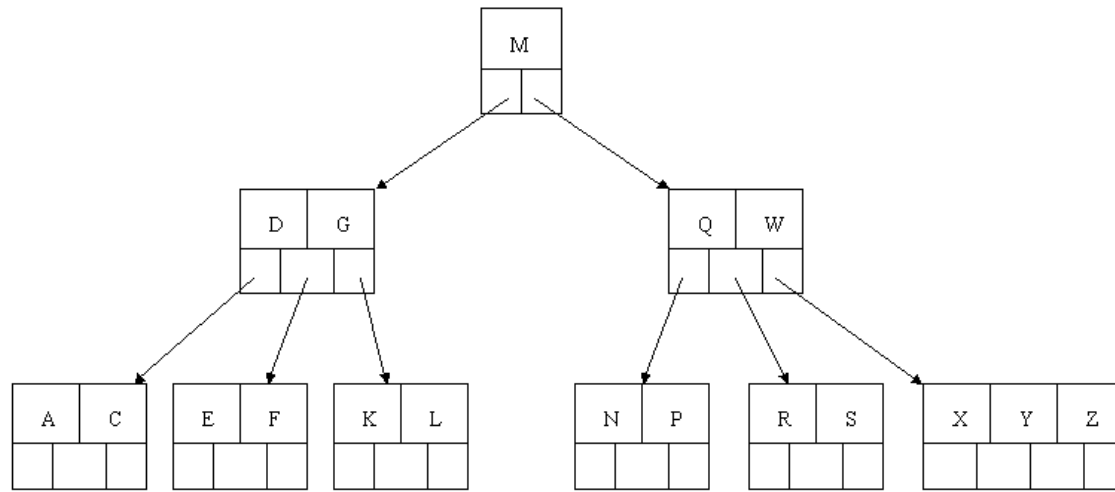
- Izbrisati ključ H (pravilo 1.1)



- Izbrisati ključ T (pravilo 2). Svodi se na brisanje, iz lista, neposrednog sledbenika u uređenju, W (“penje” se na mesto T)

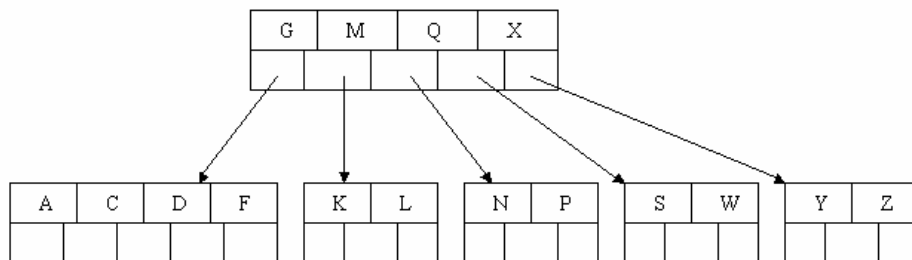
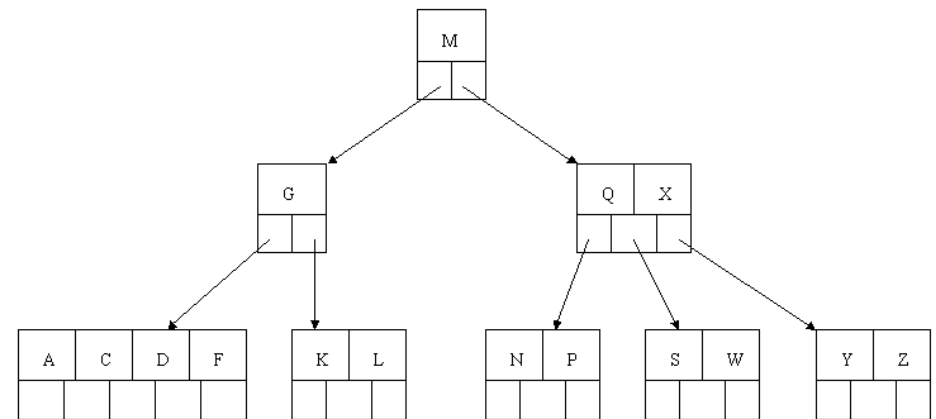
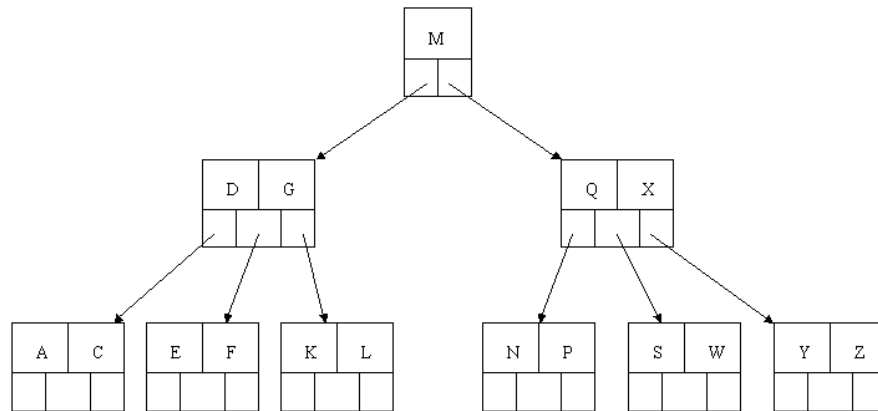


- Izbrisati ključ R (pravilo 1.2.1). Svodi se na “rotaciju”, tj. prelivanje iz desnog suseda (X, “penje se”) preko roditeljskog ključa W (spušta se)



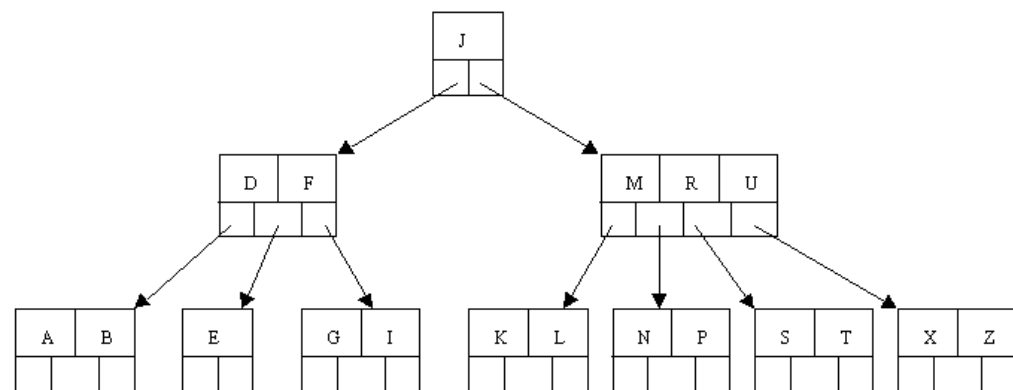
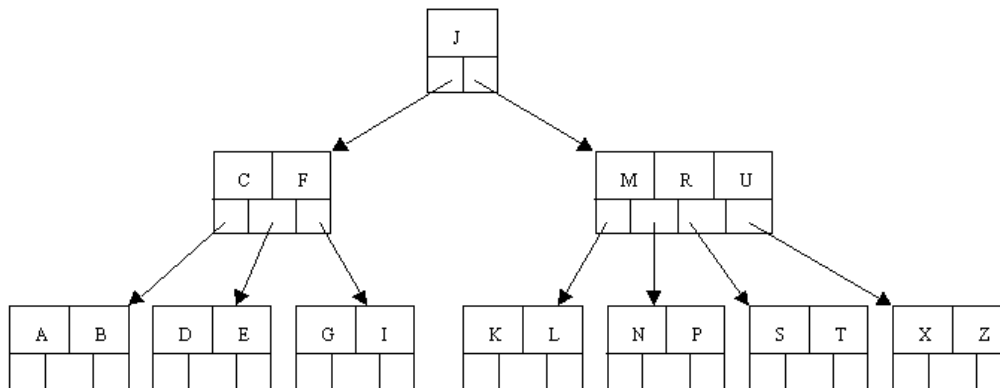


- Izbrisati ključ E (pravilo 1.2.2). Čvor se spaja sa levim bratom uz dodavanje ključa D iz roditeljskog čvora. Unutrašnji čvor sa ključem G spaja se sa bratom uz “spuštanje” ključa M. Visina drveta se smanjuje za 1.

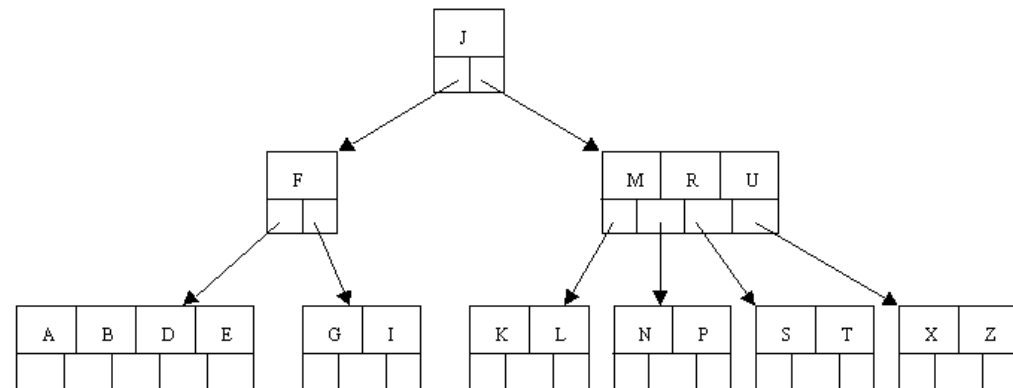
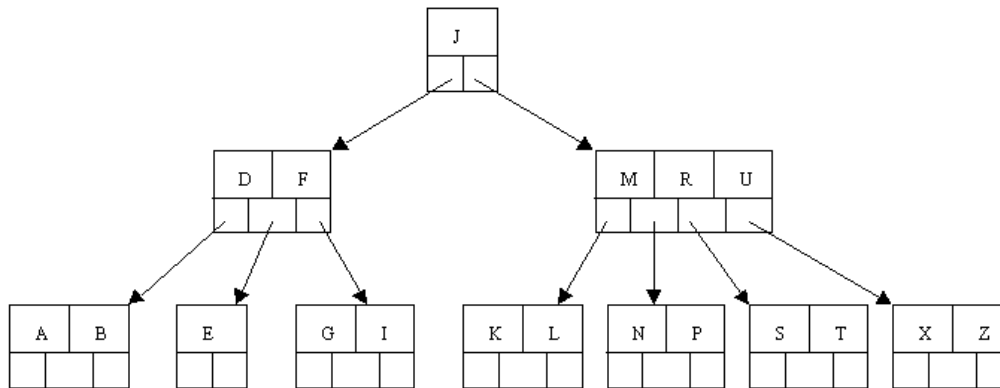


# B-stablo – primer 2.

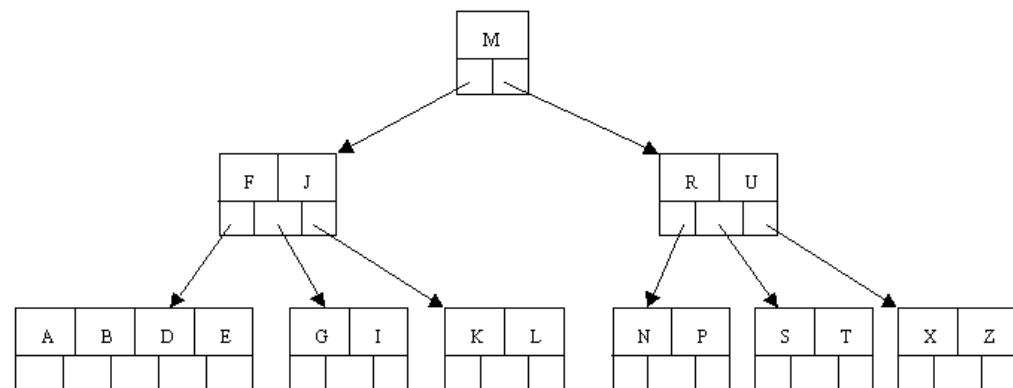
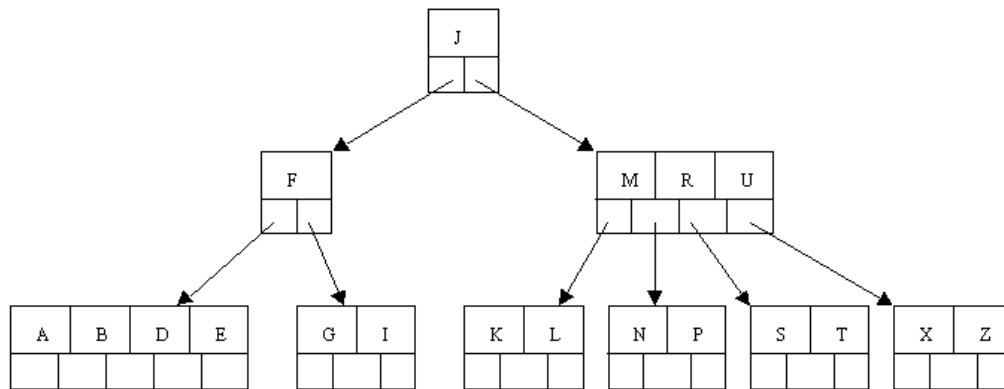
- Izbrisati ključ C – korak 1 (pravilo 2):



- Korak 2 (pravilo 1.2.2)



- Korak 3 (pravilo 1.2.2 – nast.). Čvor sa ključevima K, L promenio je prethodnika – i braću!



- <http://www.csanimated.com/animation.php?t=B-tree>
- <http://wiki.algoviz.org/AlgovizWiki/B-Trees>
- [http://www.cs.hut.fi/Research/TRAKLA2/exercises/BTree\\_insert.html](http://www.cs.hut.fi/Research/TRAKLA2/exercises/BTree_insert.html)

# B\*-stablo

- Cepanje pune stranice u B-stablu je skupa operacija
- Umesto cepanja – “prelivanje” u susedne čvorove - braću
- B\*-stablo pretrage reda  $m$  je balansirano stablo pretrage reda  $m$  za koje važi:
  - Svaki čvor, osim korena i listova, ima najmanje  $(2m-1)/3$  neposrednih sledbenika (podstabala), što podiže popunjenost na najmanje  $2/3$
  - Koren ima najmanje 2 neposredna sledbenika
  - Čvor koji nije list ima  $k$  neposrednih sledbenika i  $k-1$  ključeva

# B\*-stablo - operacije

- Pretraživanje ključa je isto kao kod B-stabla
- Unošenje (uvek u list) – analogno unošenju u B-stablo, OSIM što:
  - Ako je list pun, pokušava se *prelivanje* desnim ili levim bratom:
    - urede se svi ključevi iz lista, izabranog brata, razdvojni ključ (iz neposrednog prethodnika) i ključ koji se unosi (ukupno  $m-1+k+1+1$  ključeva)
    - $\text{floor}((m+k+1)/2)$  ključeva (manjih ili većih, u zavisnosti od toga da li se preliva iz desnog ili levog brata) ostaje na starom listu, a ostali idu u desni / levi brat (središnji se “penje”)
  - Ako prelivanje ne uspe, cepaju se 2 čvora (i puni list i njegov brat) na 3 čvora, tako što u prvi ide  $\text{floor}((2m-2)/3)$ , u drugi  $\text{floor}((2m-1)/3)$  a u treći  $\text{floor}((2m)/3)$  ključeva, a u čvor neposredni prethodnik idu dva razdvojna ključa
  - Prethodno pravilo se propagira po potrebi sve do korena

# B\*-stablo - operacije

- Brisanje: spajanje 3 čvora u 2 (ako je potrebno)
- Bolja popunjenost, manja visina i efikasnije operacije u odnosu na B-stabla



$B^*$ -stablo - primer