

**Strukture podataka**

**Indeksne  
strukture**

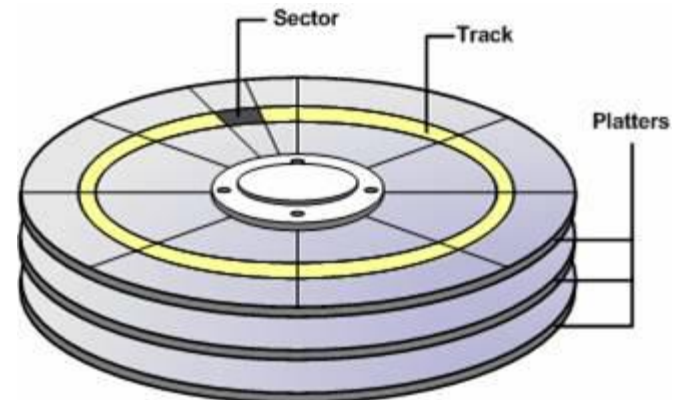
# Organizacija podataka na hard disku



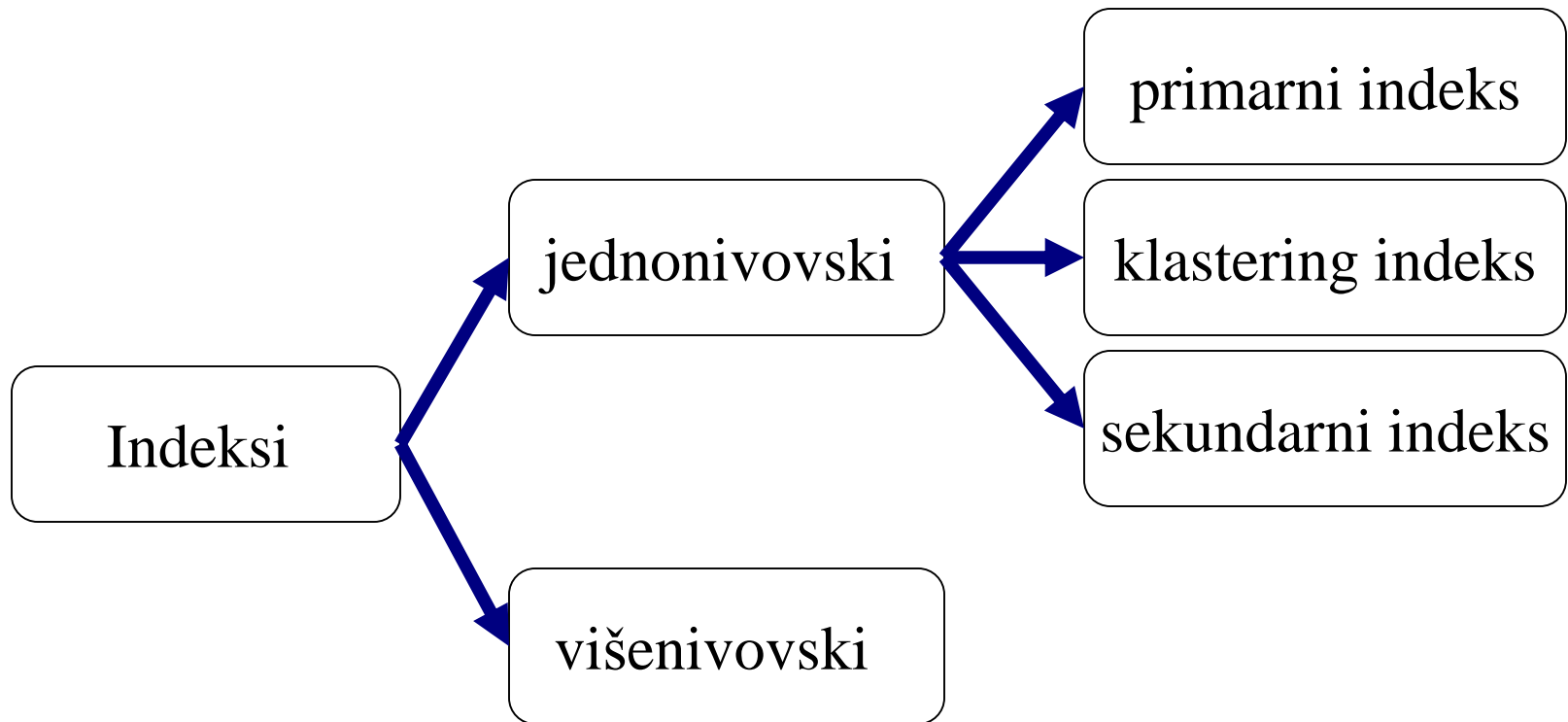
```
Command Prompt
C:\Temp>diskpart -i 1
----- Drive 1 Geometry Information -----
Cylinders = 261
TracksPerCylinder = 255
SectorsPerTrack = 63
BytesPerSector = 512
DiskSize = 2146798080 (Bytes) = 2047 (MB)

----- Drive Partition 0 Information -----
StartingOffset = 32256
PartitionLength = 2138540544
HiddenSectors = 63
PartitionNumber = 1
PartitionType = 6

End of partition information. Total existing partitions: 1
C:\>
```

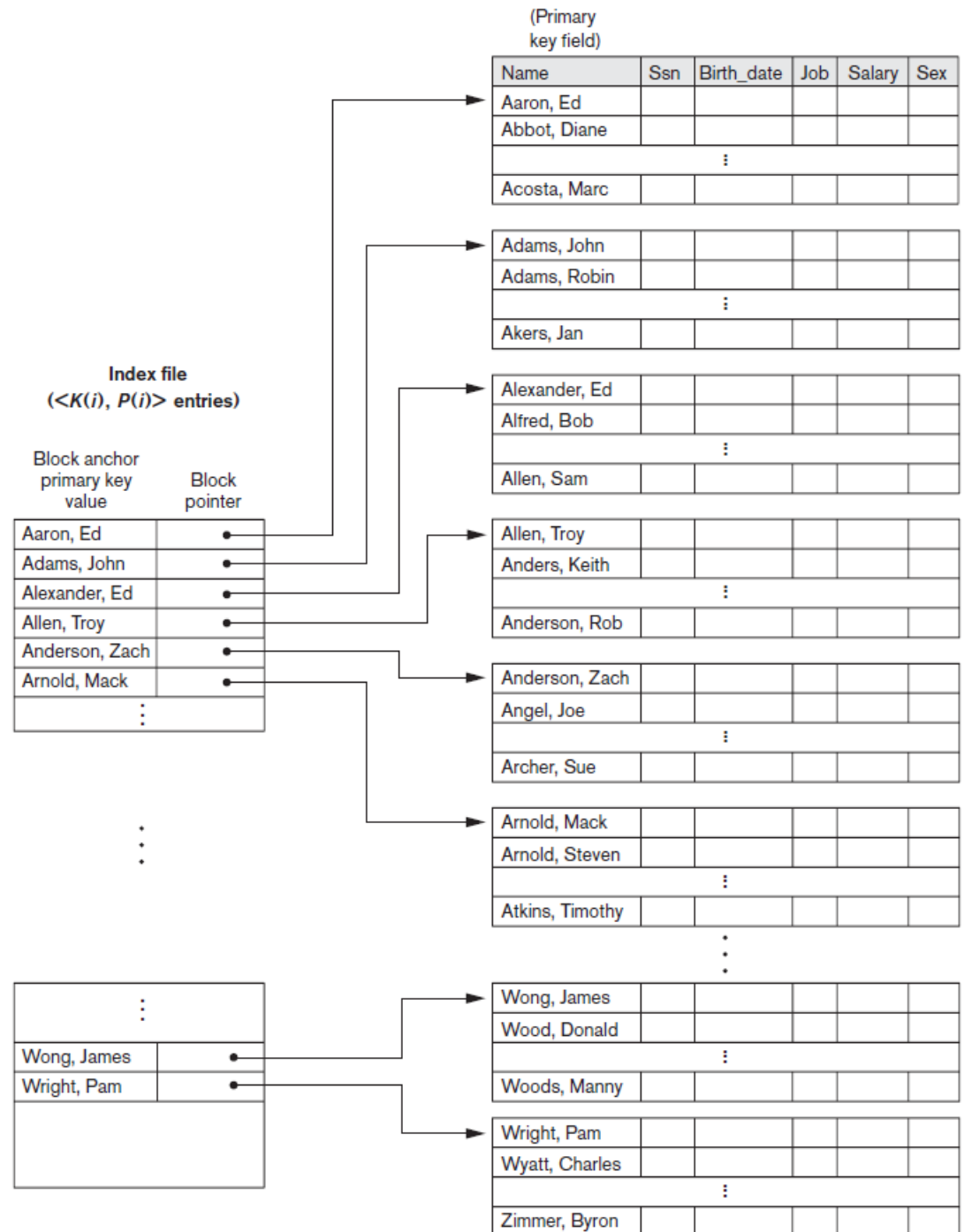


# Indeksne strukture datoteka



# Primarni indeks

Može se primeniti samo ako je datoteka uređena po ključnom polju



# Br. pristupa kod dat. sa uređenim poljem ključa

## a) BEZ INDEKSA

Neka je veličina datoteke  $r = 30.000$  slogova, svaki slog po  $R_1 = 100$  [B], a svaki blok po  $B_1 = 1024$  [B].

Faktor blokiranja:

$$fb_1 = \left\lfloor \frac{B_1}{R_1} \right\rfloor = \left\lfloor \frac{1024}{100} \right\rfloor = 10 \quad \text{sloga/bloku}$$

Br. blokova:

$$bb_1 = \left\lceil \frac{30.000}{10} \right\rceil = 3.000 \quad \text{blokova (za celu datoteku)}$$

Za traženje sloga korišćenjem binarnog traženja:

$$\lceil \log_2 bb_1 \rceil = \lceil \log_2 3.000 \rceil = 12 \quad \text{pristupa disku}$$

## b) SA KORIŠĆENJEM PRIMARNOG INDEKSA

Neka je ključ dužine 9[B] a pokazivač 6[B]

Veličina sloga indeksa:  $R_2 = 9 + 6 = 15$  B

$$fb_2 = \left\lfloor \frac{1024}{15} \right\rfloor = 68 \quad \text{sloga/bloku}$$

Po jedan slog za svaki blok u datoteci

$$bb_2 = \left\lceil \frac{bb_1}{fb_2} \right\rceil = \left\lceil \frac{3.000}{68} \right\rceil = 45 \quad \text{blokova (za primarni indeks)}$$

Pristup bloku sa zadatim ključem (binarno traženje)

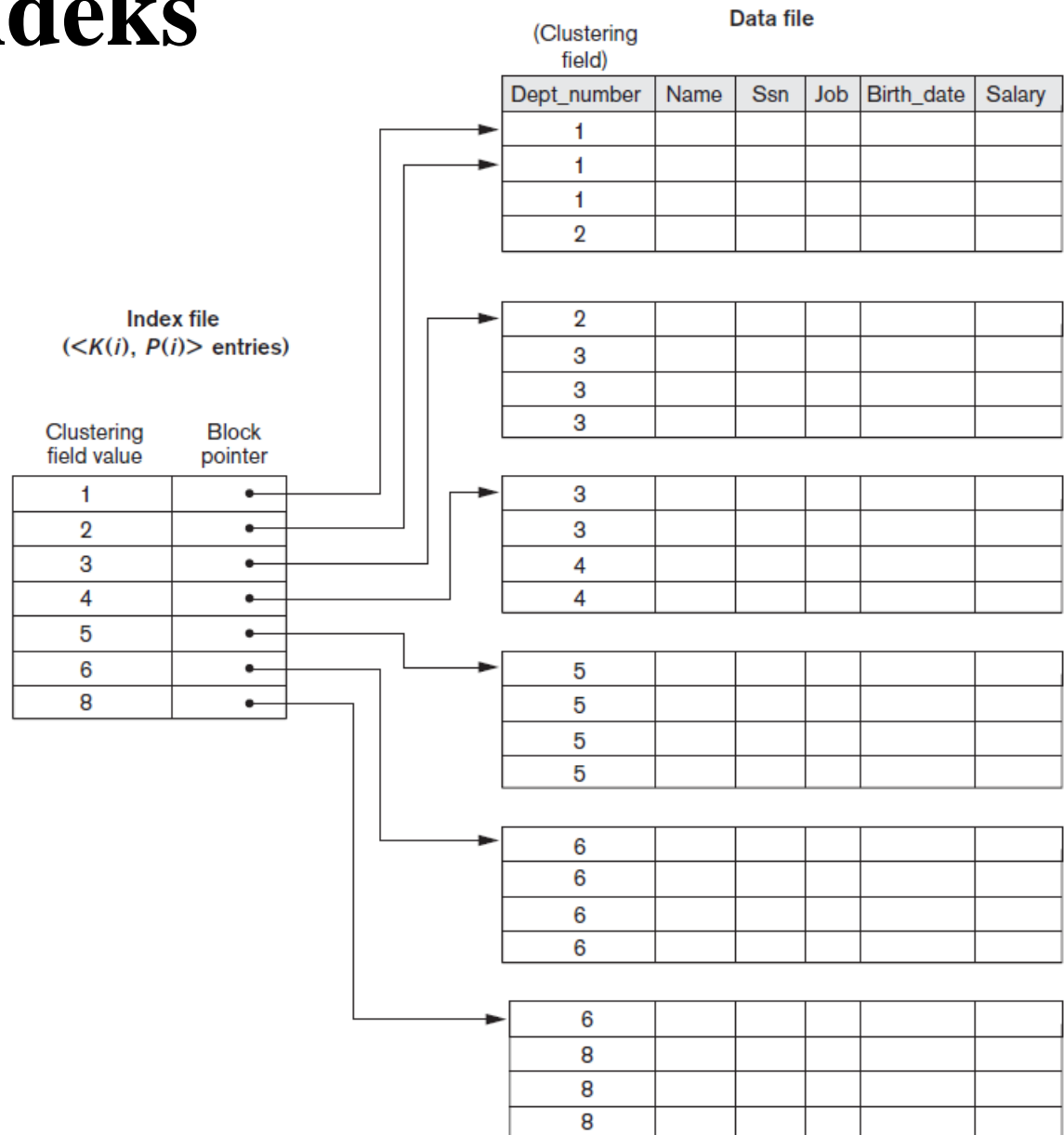
$$\lceil \log_2 bb_2 \rceil + 1 = \lceil \log_2 45 \rceil + 1 = 6 + 1 = 7 \quad \text{pristupa disku}$$

↑  
Pristup indeksnom (slogu) bloku

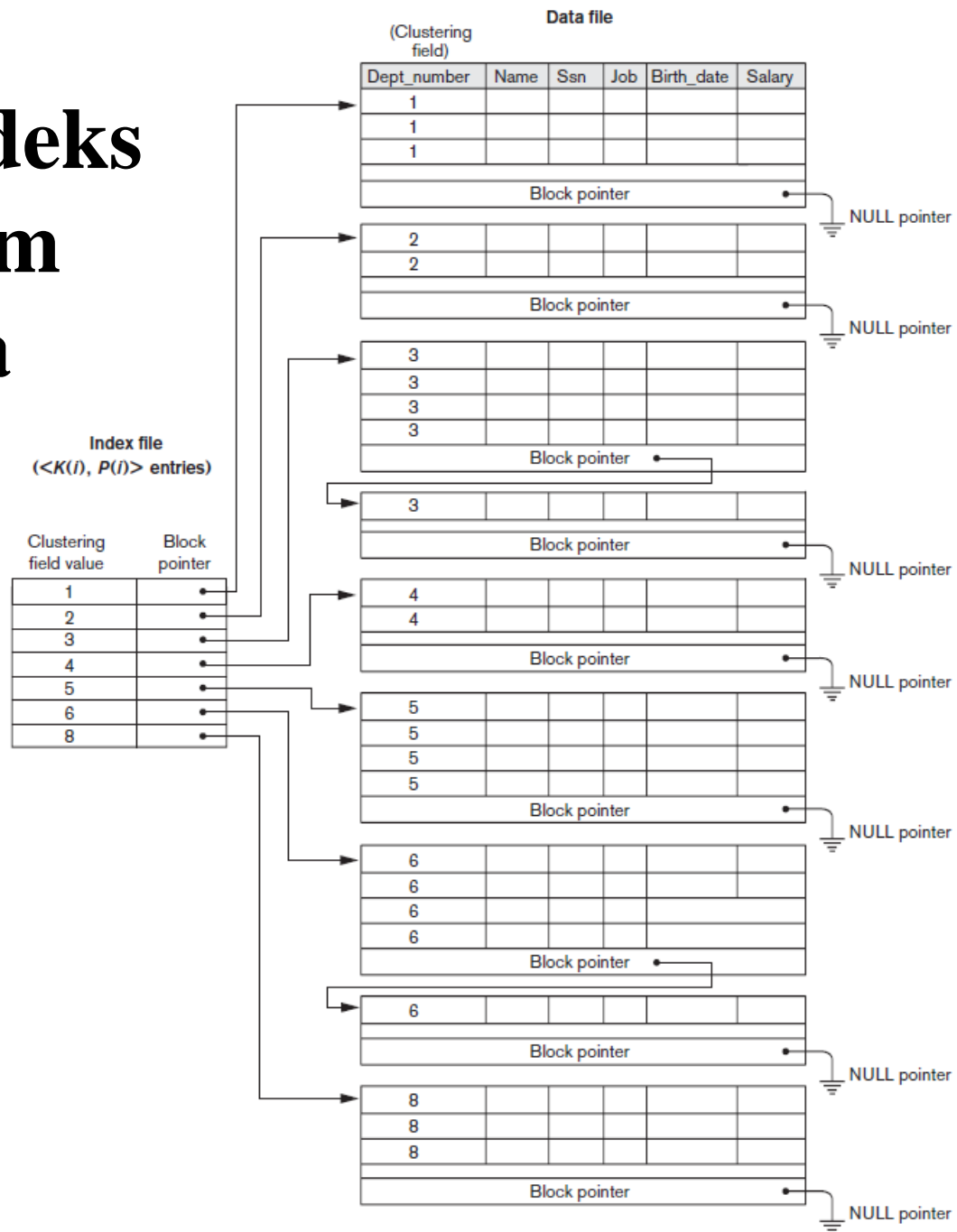
↑  
Pristup bloku u glavnoj datoteci na osnovu indeksa

# Klastering indeks

Koristi se za indeksiranje datoteke koja je uređena po neključnom polu



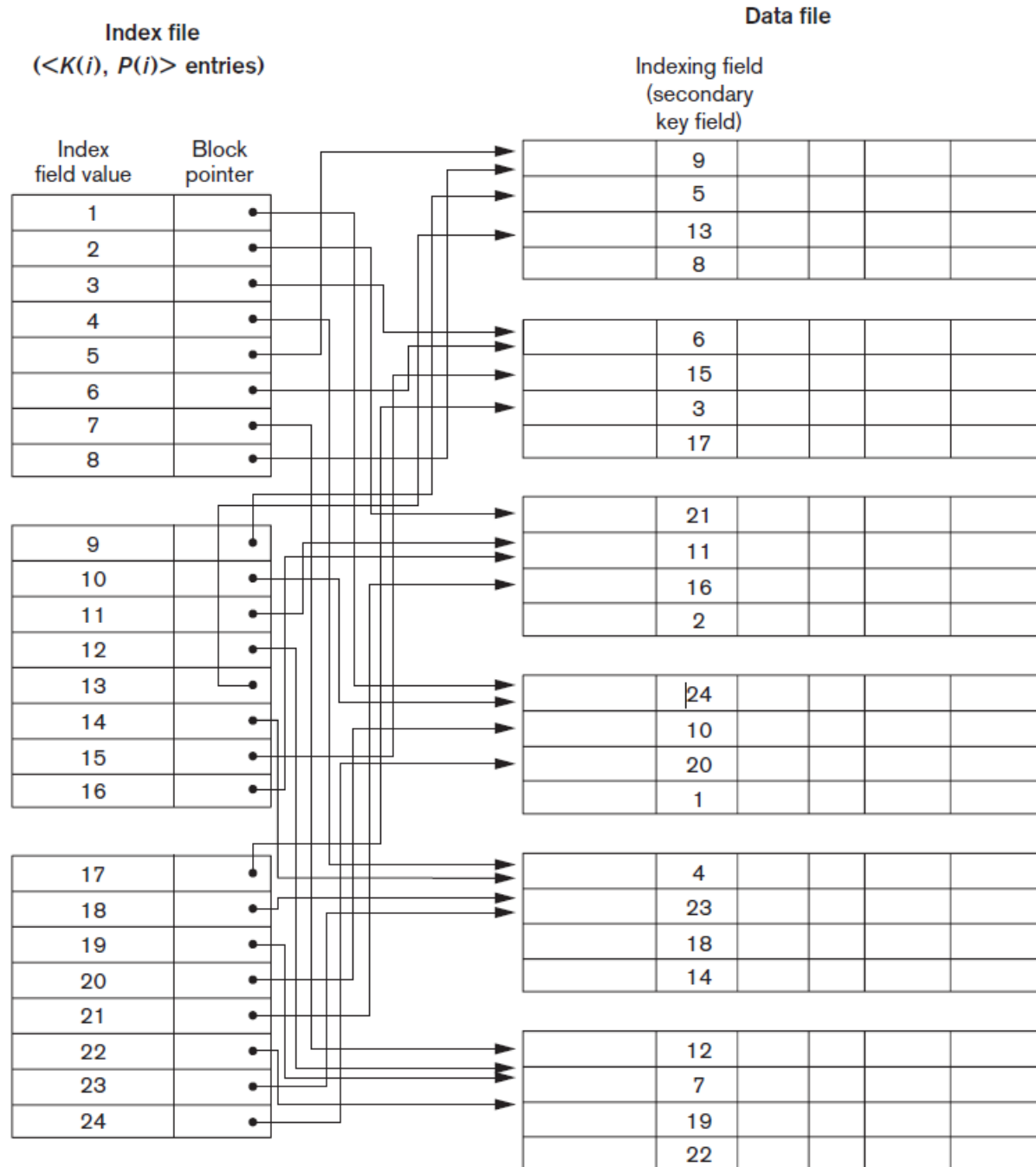
# Klastering indeks sa odvojenim blokovima





# Gusti sekundarni indeks

Koristi se kada datoteka nije  
uređena po ključnom polju



# Br. pristupa kod dat. sa neuređenim poljem ključa

## a) BEZ SEKUNDARNOG INDEKSA

Traženje je linearno, u najgorem slučaju, za prethodni primer (3000 blok.) potrebno je 3000 pristupa, a u srednjem 1500.

## b) SA SEKUNDARNIM INDEKSOM

$$R_2 = 15 \text{ B}$$

$$fb_2 = \left\lfloor \frac{1024}{15} \right\rfloor = 68 \quad \text{sloga/bloku}$$

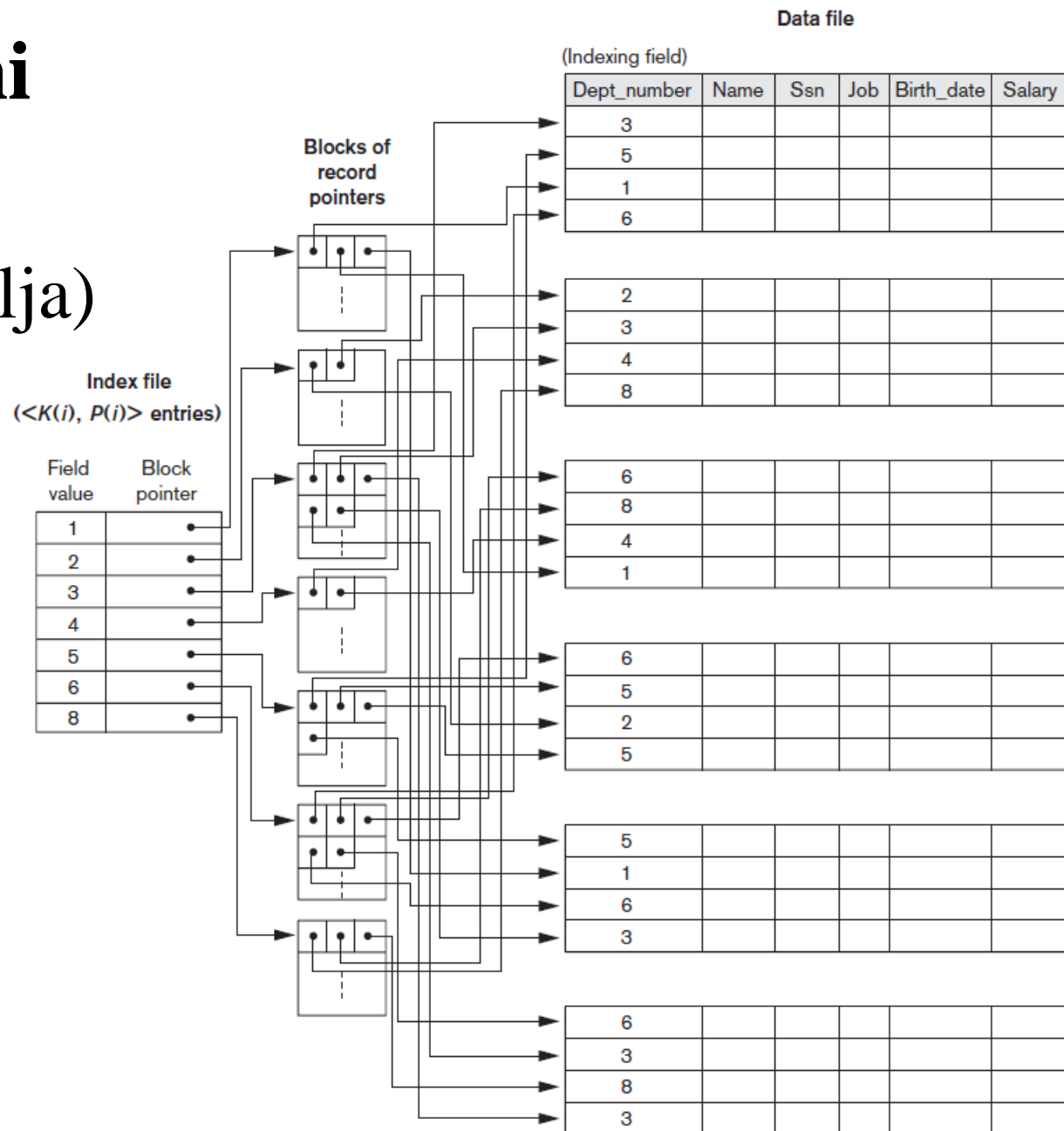
$$bb_2 = \left\lceil \frac{bs}{R_2} \right\rceil = \left\lceil \frac{30.000}{68} \right\rceil = 442 \quad \text{bloka}$$

Binarnim traženjem (indeks je uvek uređen) br. pristupa disku je:

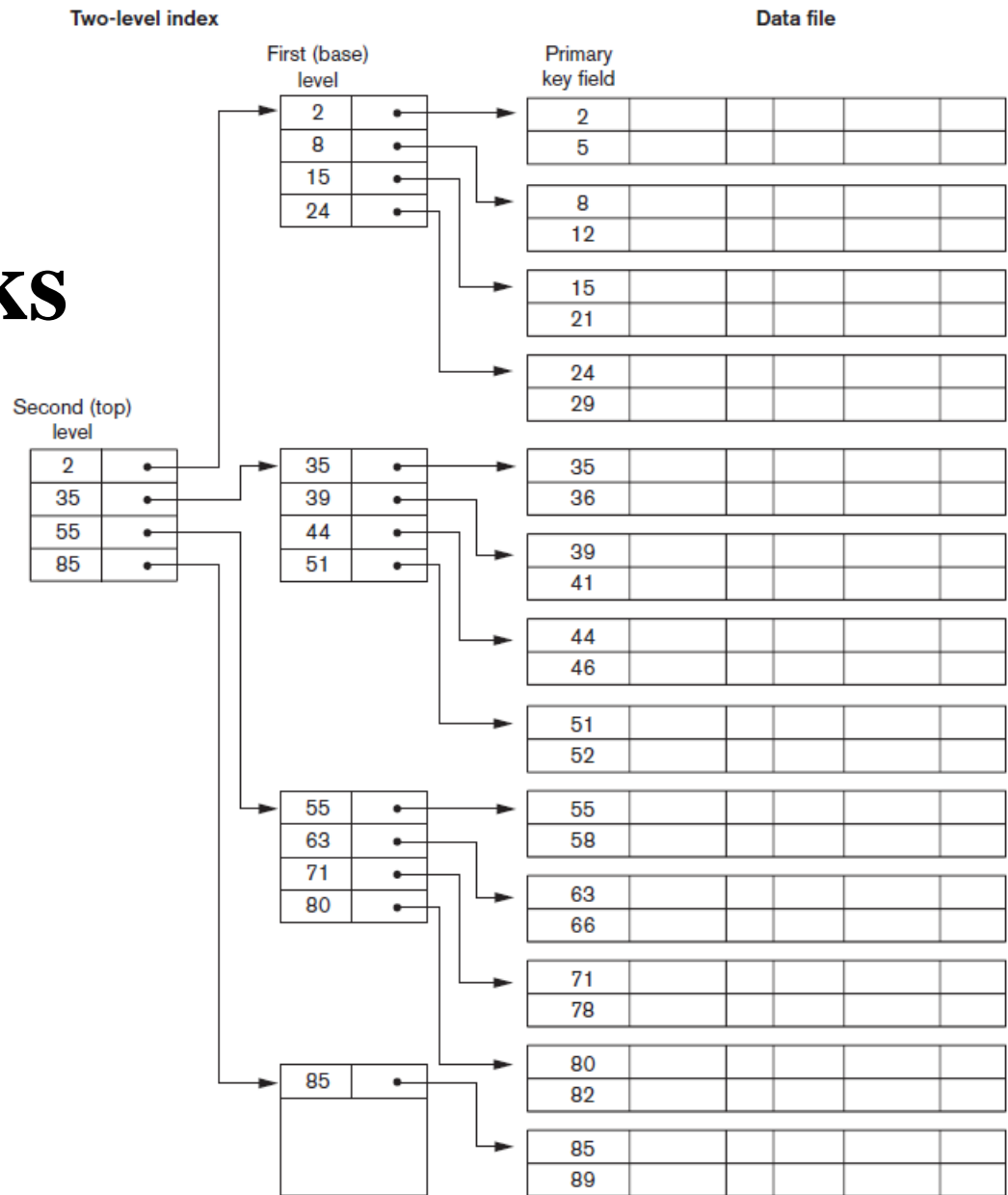
$$\lceil \log_2 bb_2 \rceil + 1 = \lceil \log_2 442 \rceil + 1 = 9 + 1 = 10 \quad \text{Ubrzanje 150x !!!}$$

# Sekundarni indeks

(neključnog polja)



# Dvonivovski primarni indeks



# Ubrzanje pristupa korišćenjem višenivovskog indeksiranja

To je uvođenje indeksiranja za indeksnu datoteku, tj. max.ubranje pristupa formiranjem stabla indeksnih slogova. Ako je br. slogova indeksa na prvom nivou  $r_1$ , a faktor blokiranja (br.slogova po bloku)  $fb_1$ , tada je br.slogova na drugom nivou:

$$r_2 = \left\lceil \frac{r_1}{fb_1} \right\rceil \quad \text{na trećem:} \quad r_3 = \left\lceil \frac{r_2}{fb_2} \right\rceil \quad \text{itd.} \quad r_{i+1} = \left\lceil \frac{r_i}{fb_i} \right\rceil$$

ukoliko je faktor blokiranja isti za sve nivoe indeksa (što je najčešće) i označimo ga sa  $f_0$

$$r_{i+1} = \left\lceil \frac{r_i}{f_0} \right\rceil = \left\lceil \frac{r_{i-1}}{f_0^2} \right\rceil = \dots = \left\lceil \frac{r_1}{f_0^{i+1}} \right\rceil$$

nivoi se dodaju sve dok u poslednjem ne ostane samo 1 blok:

$$1 \geq \frac{r_1}{f_0^t} \Rightarrow t = \left\lceil \log_{f_0} r_1 \right\rceil \quad - \text{ br.nivoa}$$

# Primer izračunavanja broja nivoa i faktor ubrzanja pristupa

Za podatke iz prethodnog primera:

$fb_1 = f_0 = 68$  sloga/bloku

$bb_1 = 442$  bloka na prvom nivou

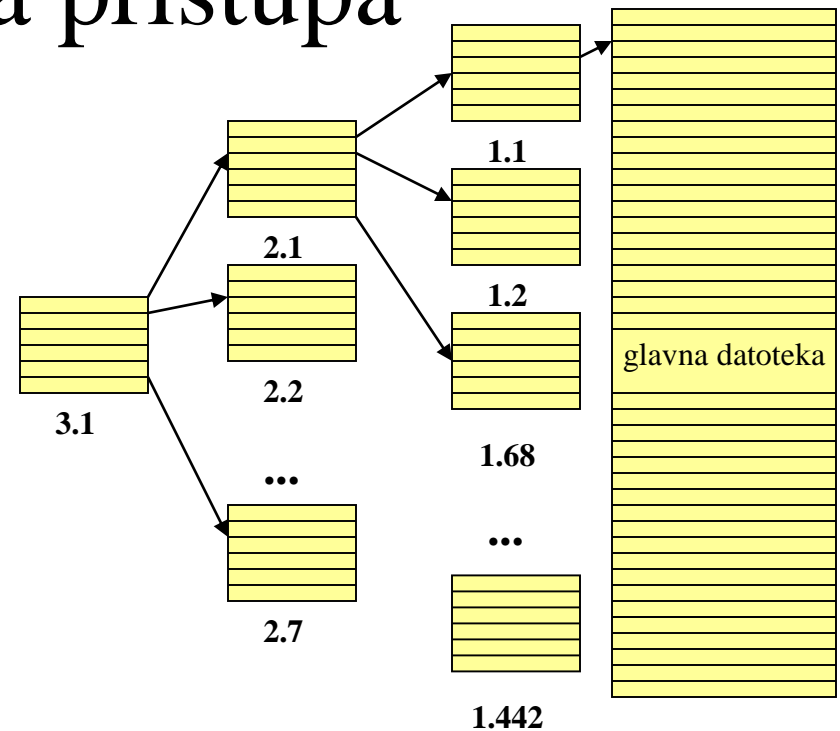
Korišćenjem indeksiranja u više nivoa:

- za drugi nivo:

$$b_2 = \left\lceil \frac{b_1}{f_0} \right\rceil = \left\lceil \frac{442}{68} \right\rceil = 7 \text{ blokova}$$

- za treći nivo:

$$b_3 = \left\lceil \frac{b_2}{f_0} \right\rceil = \left\lceil \frac{7}{68} \right\rceil = 1 \text{ blok}$$



Br.pristupa:

$$t + 1 = 3 + 1 = 4$$

2.5x brže

br.nivoa indeksa

# Zadaci

- Podaci o studentima Elektronskog fakulteta smešteni su u jednu datoteku. Neka fakultet ima 800 studenata, pri čemu se za svakog studenta pamte: br.indexa (4B), prezime (15B), ime oca (15B), ime (15B), smer (1B), godina studija (1B), godina rođenja (4B), godina koje je upisan fakultet (4B). Ako je veličina bloka na disk 512B, a veličina pokazivača na blok na disku 6B:
  - predložiti i skicirati strukturu za ubrzavanje pristupa (indeksiranje) na osnovu smeru,
  - predložiti i skicirati strukturu za indeksiranje na osnovu broja indeksa, ukoliko je datoteka uređena po ovom polju,
  - predložiti i skicirati strukturu za indeksiranje na osnovu broja indeksa, ukoliko je datoteka NIJE uređena po ovom polju,
  - predložiti i skicirati strukturu za indeksiranje na osnovu prezimena,
  - predložiti i skicirati strukturu za indeksiranje na osnovu godine studija.
- Za svaku od indeksnih struktura odrediti veličinu (za koliko procenata je veća datoteka ako se uvede indeksiranje) i koliko puta se ubrzava pristup.

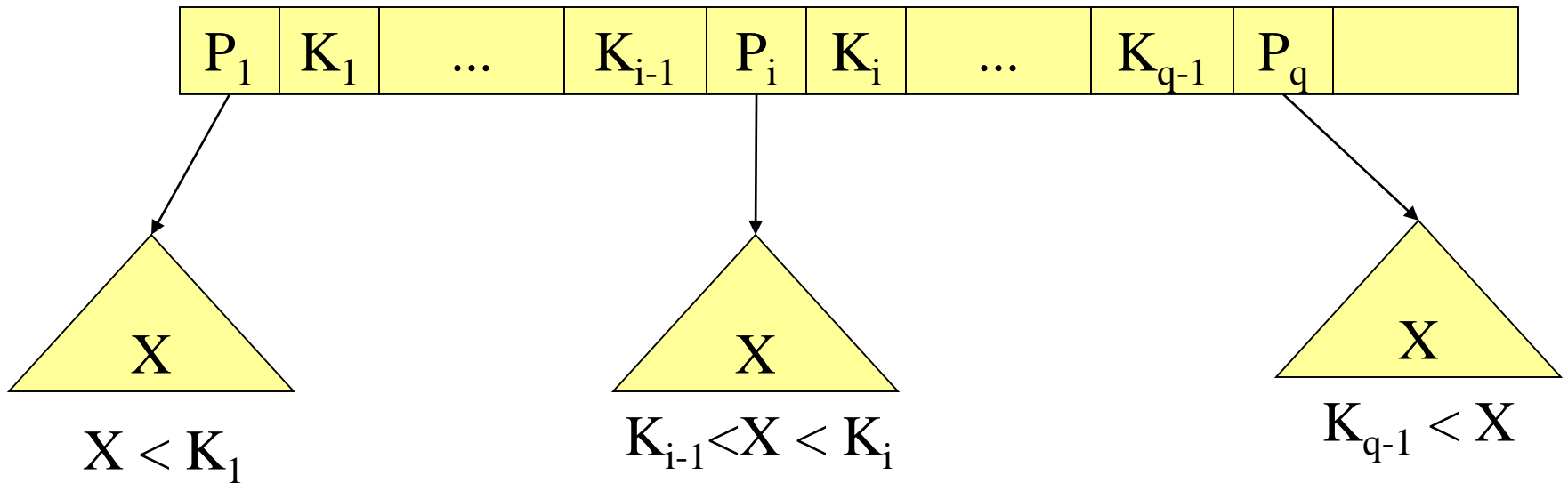
# Strukture podataka

**Strabla traženja po  
M putanja**



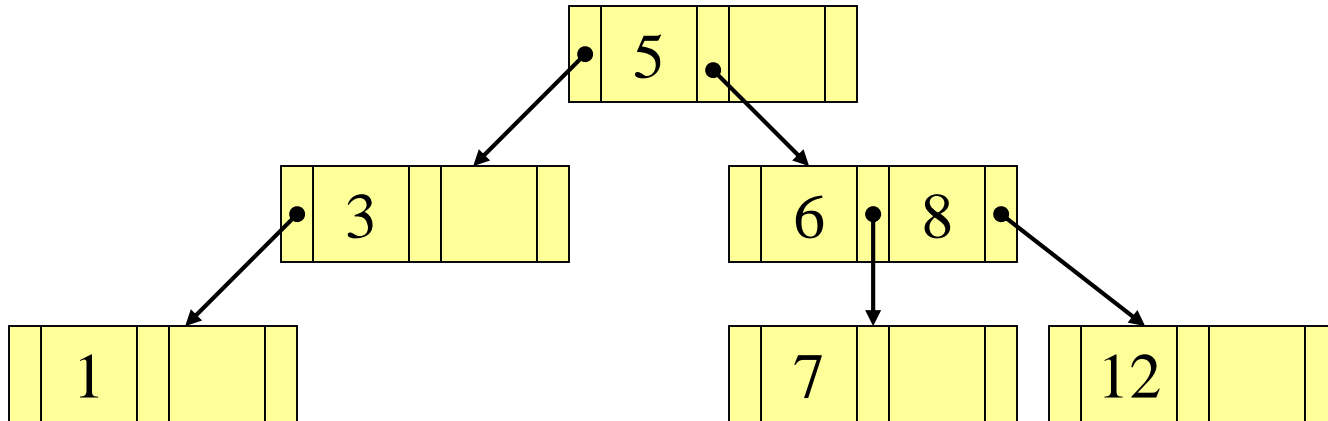
# Stablo traženja

reda **Q**



# Primer

## Stablo reda 3



# Definicija

- *Stablo traženja po  $\mathcal{M}$  putanja je stablo traženja čiji svaki unutrašnji čvor sadrži  $Q$  podstabla i  $Q-1$  ključ, pri čemu je  $2 \leq Q \leq \mathcal{M}$  ( $\mathcal{M} \geq 2$ ).*

# Kako izabrati M?

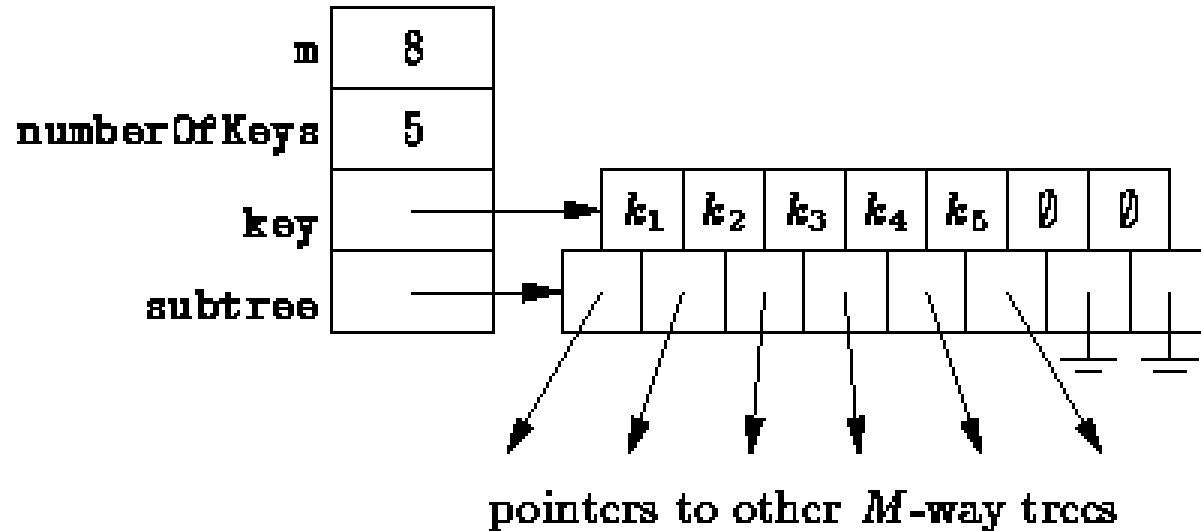
Pristup disku iznosi 1-10ms, a memoriji 10-100ns, dakle  $10^5$ - $10^6$  puta brže. Da bi se maksimizirale performanse treba minimizirati br.pristupa disku.

Veličina bloka na disku je 512 - 4096 B.

M treba izabrati tako da čvor zauzima celi blok.

# Memorijska reprezentacija

---



# Struktura podataka

## B-Stabla

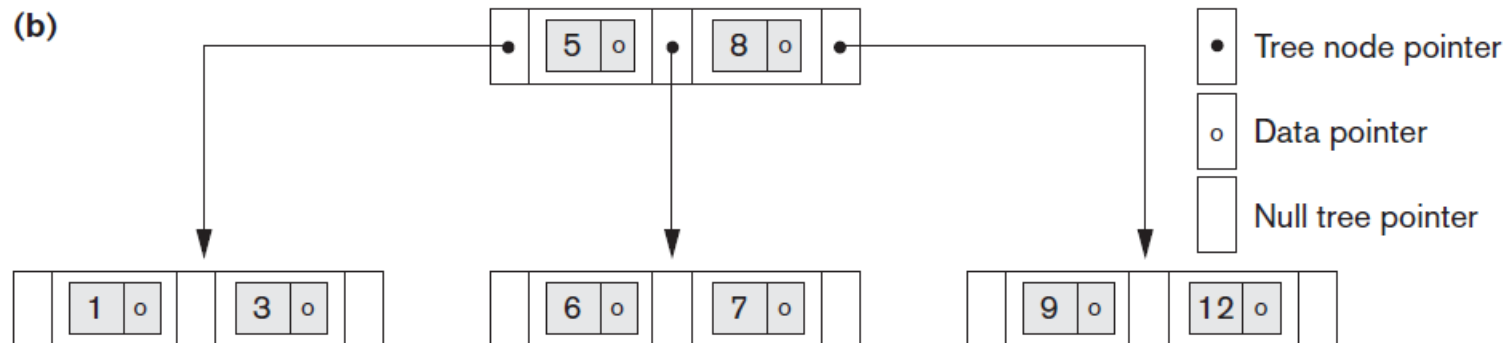
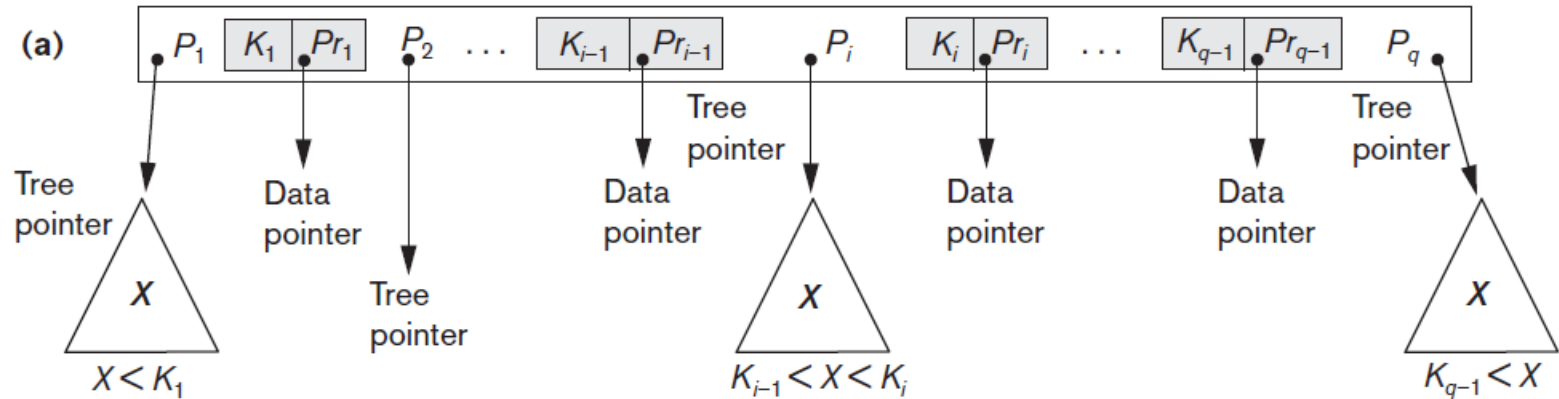
# Definicija

*B-stablo reda  $M$  je stablo traženja po  $M$  putanja za koje važi:*

- koren ima najmanje 2 a najviše  $M$  podstabala*
- svi unutrašnji čvorovi imaju između  $\lceil M/2 \rceil$  i  $M$  podstabala*
- svi spoljnji čvorovi (listovi) su na istom nivou.*

B-stabla su balansirana stabla traženja po  $M$  putanja, čiji su svi unutrašnji čvorovi bar 50% popunjeni.

# Primer B-stabla





# Umetanje

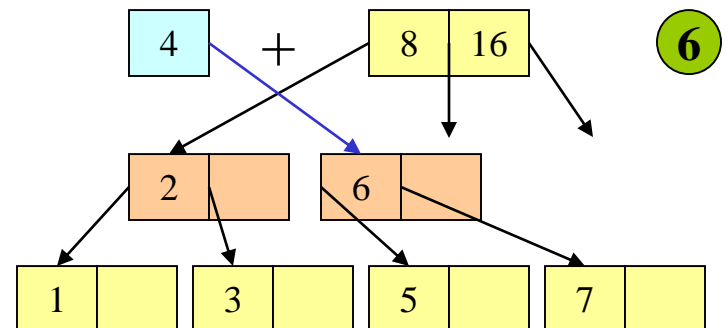
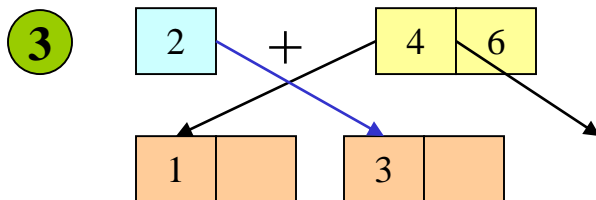
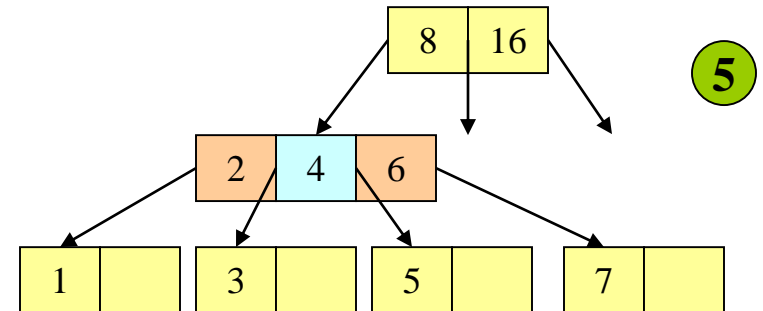
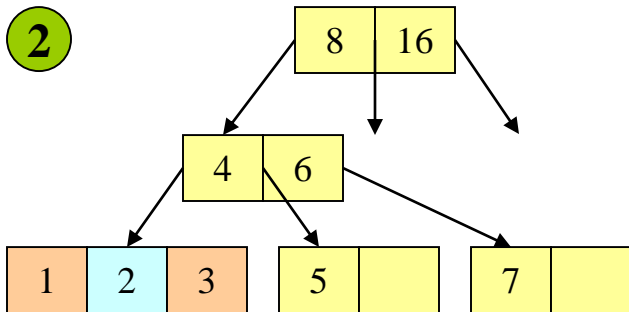
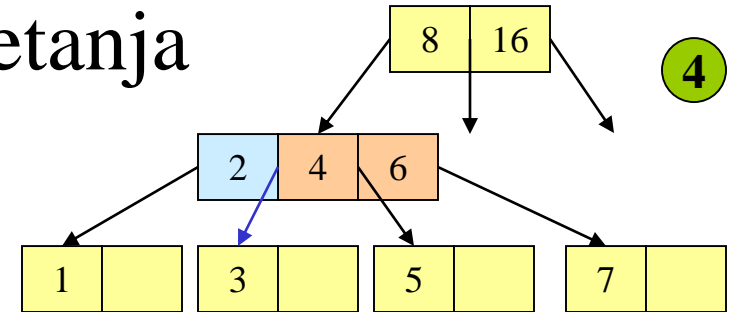
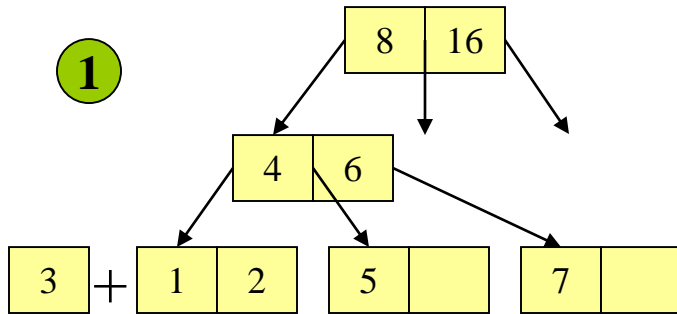
Da bi se održala struktura B-stabla, novi ključ umećemo uvek u odgovarajući list, i nikada ne umećemo samo ključ X, već i pokazivač na podstablo, tj. par (x,0), jer je podstablo inicijalno prazno.

$$T = \{T_0, k_1, T_1, k_2, T_2, \dots, k_{n-1}, T_{n-1}\}. \quad k_i < x < k_{i+1}$$

$$T' = \{T_0, k_1, T_1, k_2, T_2, \dots, k_i, T_i, x, \emptyset, k_{i+1}, T_{i+1}, \dots, k_{n-1}, T_{n-1}\}.$$

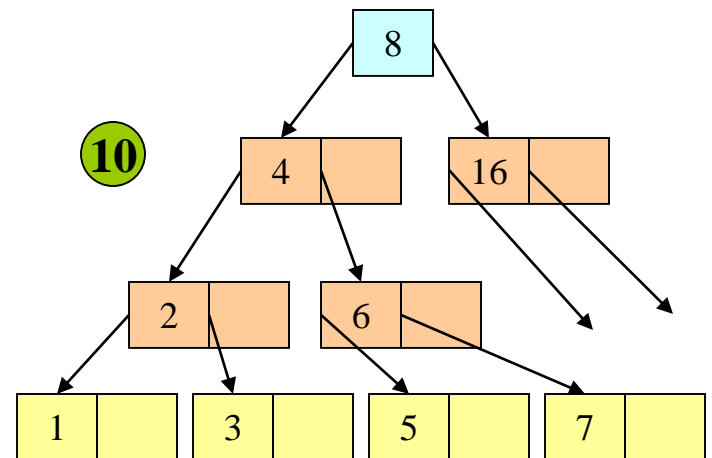
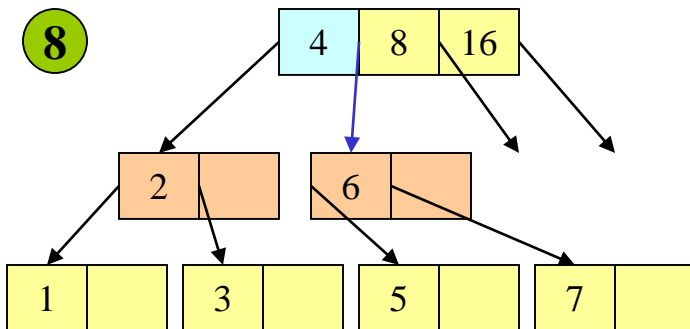
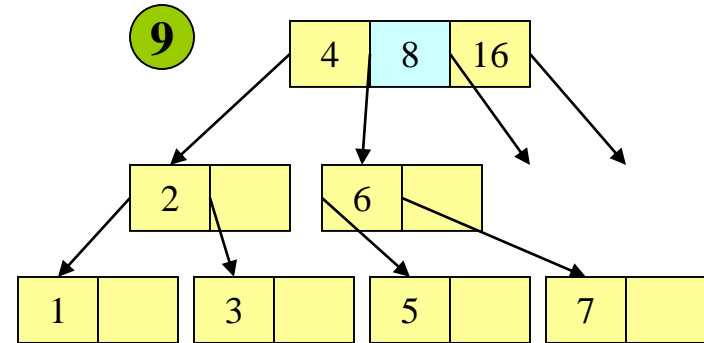
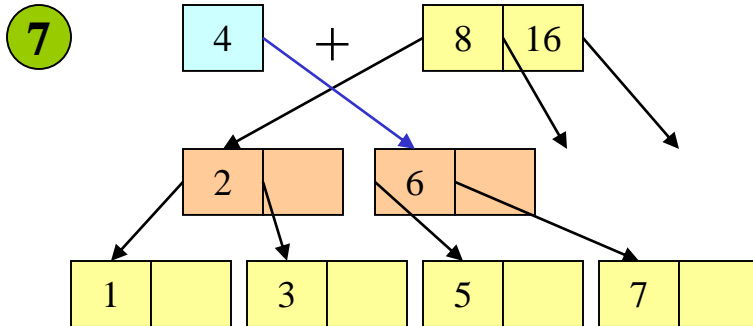
Ako čvor (list) u koji se dodaje nije popunjen (br. ključeva manji od M-1), pravi se mesto za novi ključ i podstablo pomeranjem udesno, a u protivnom čvoru se dodaje novi ključ, a zatim se čvor deli na dva čvora. Maksimalni ključ iz levog čvora (zajedno sa pokazivačem na desni čvor) seli se u roditeljski. Ako je roditeljski čvor popunjen i on se deli na dva čvora. Propagacija “cepanja” čvorova može se nastaviti do korena i ukoliko je i on popunjen stablo povećava svoju visinu.

# Primeri umetanja



Srednji ključ i pokazivač na desni čvor nastao cepanjem tekućeg umeću se u roditeljski čvor

# Primeri umetanja



# Umetanje

Algoritam umetanja je dvoprolazni:

1. od korena ka listovima da bi se našao list u koji se umeće novi ključ (*insert* metod) i
2. od listova ka korenu uz “cepanje” čvorova i umetanje novih ključeva u roditeljske čvorove (*insertPair* metod).

# Brisanje

## 1. **Brisanje iz lista**

**1.1.** Ako je nakon brisanja čvor bar pola popunjen, svi ključevi i pokazivači desno od obrisnog pomeraju se ulevo za jedno mesto.

**1.2.** Ako je čvor popunjen manje od pola (tj.  $\lceil m/2 \rceil - 1$ ) nastaje potkoračenje

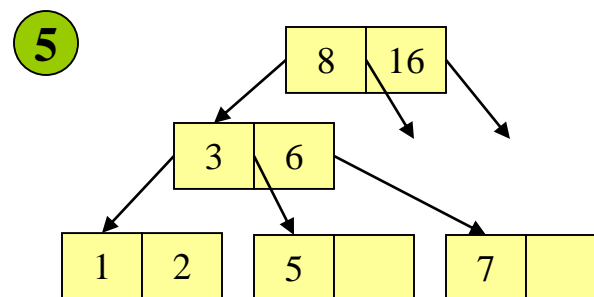
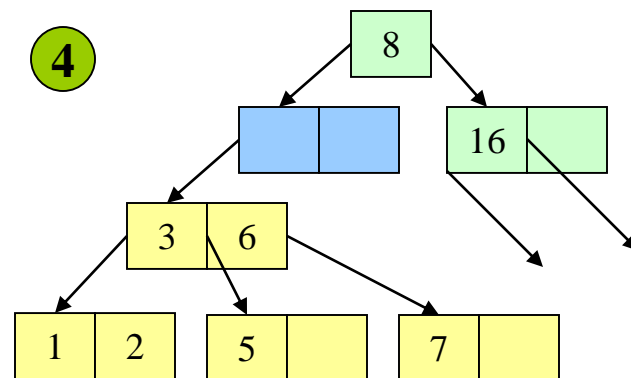
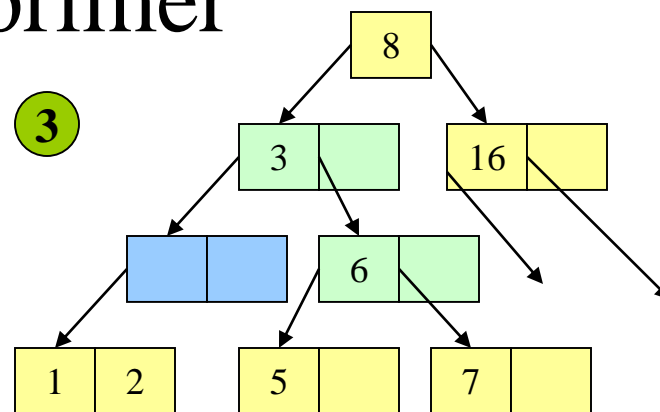
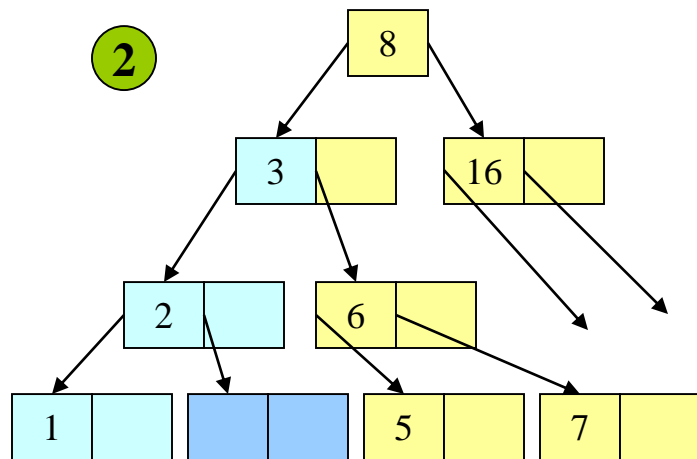
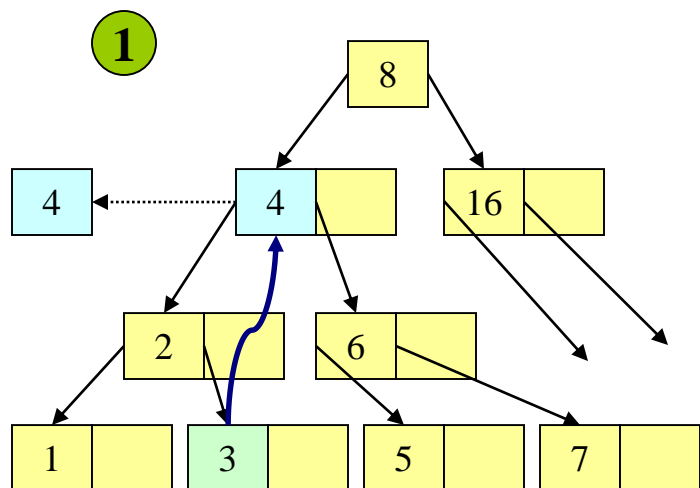
**1.2.1.** Ako postoji levo ili desno čvor na istom nivou sa više od  $\lceil m/2 \rceil - 1$  ključeva, tada se mešaju ta dva čvora uz dodatak ključa iz roditeljskog čvora koji deli ta dva čvora, a zatim se rezultujući čvor deli na dva dela i ključ na osnovu koga je ostvarena podela ide u roditeljski čvor.

**1.2.2.** Ako u susedstvu nema čvora sa više od  $\lceil m/2 \rceil - 1$  ključeva, mešaju se tekući čvor, jedan od suseda i odgovarajući ključ iz roditeljskog čvora u okviru jednog čvora, tekući čvor se briše. Ključevi i pokazivači u roditeljskom čvoru se pomeraju. Ako broj ključeva u roditeljskom čvoru padne ispod polovine nastaje potkoračenje i nastavlja se izvršenje koraka 1.2. sve dok se ne dodje do koraka 1.2.1. ili se ne dostigne koren.

**1.2.2.1.** Ako je roditeljski čvor koren samo sa jednim ključem, a mešaju se dva susedna čvora, tada čvor koji se dobija mešanjem suseda i jedinog ključa iz korena postaje novi koren, a drugi sused i stari koren se brišu.

**2. **Brisanje iz unutrašnjeg čvora.**** Obavlja se isto kao brisanje kopiranjem u uredjenom binarnom stablu. Nalazi se najdesniji ključ u levom podstablu (u B-stablu to je obavezno list) i on se kopira u odgovarajući čvor, a zatim briše iz lista.

# Brisanje - primer



# B\* stabla

- B-stabla – svi čvorovi, sem korena, moraju biti bar  $\frac{1}{2}$  puni.
- B\*-stabla – svi čvorovi, sem korena, moraju biti bar  $\frac{2}{3}$  puni, tj. mora imati  $k$  ključeva,  $\lfloor (2m-1)/3 \rfloor \leq k \leq m-1$ .
- B\*\*-stabla – svi čvorovi, sem korena, moraju biti bar  $\frac{3}{4}$  puni (75%),
- B<sup>n</sup>-stabla – svi čvorovi, sem korena, moraju biti bar  $\frac{n+1}{n+2}$  puni.

# $B^+$ - stablo

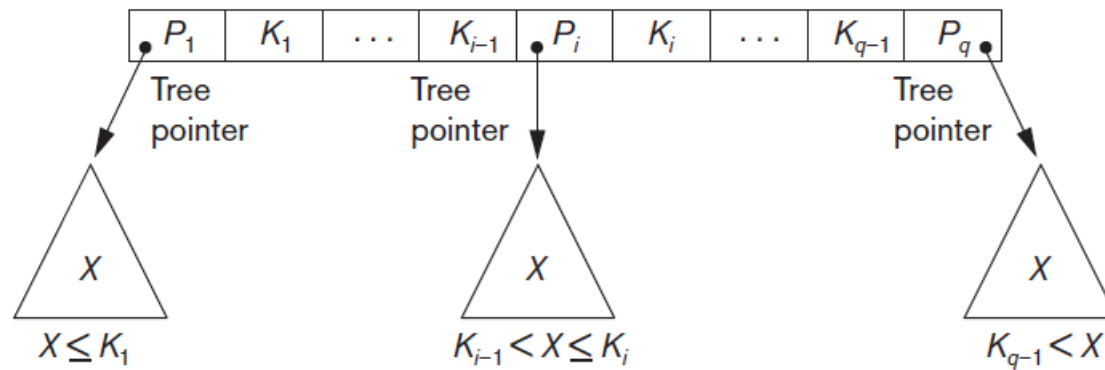
B-stabla kod kojih je omogućeno i linearni prolazak kroz čvorove (brže nego obilazak stabla), ulančavanjem listova naziva se  $B^+$ -stablo. Da bi se omogućio prolazak kroz sve podatke samo obilaskom listova, unutrašnji čvorovi sadrže kopije ključeva iz listova, ali ne i pokazivače na same podatke.

Prilikom dodavanja novog ključa, on se dodaje u list, zajedno sa pokazivačem na same podatke, a zatim, ako se javi cepanje čvora, vrednost srednjeg ključa se KOPIRA u roditeljski čvor. (Ne premešta se kao kod B-stabla!)

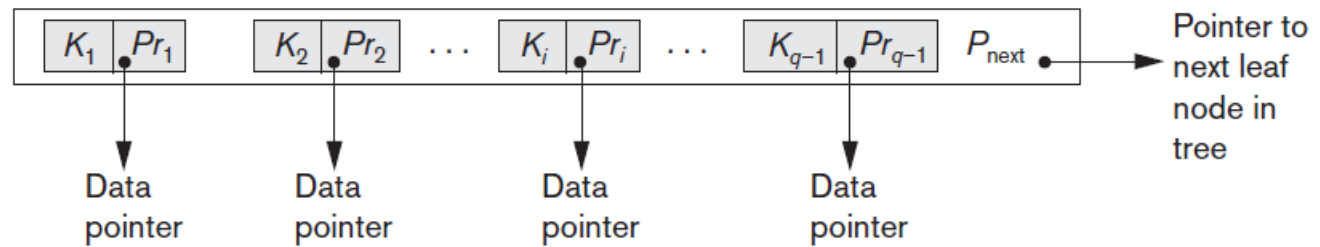


# B<sup>+</sup> - stablo

(a)



(b)



# Umetanje i brisanje

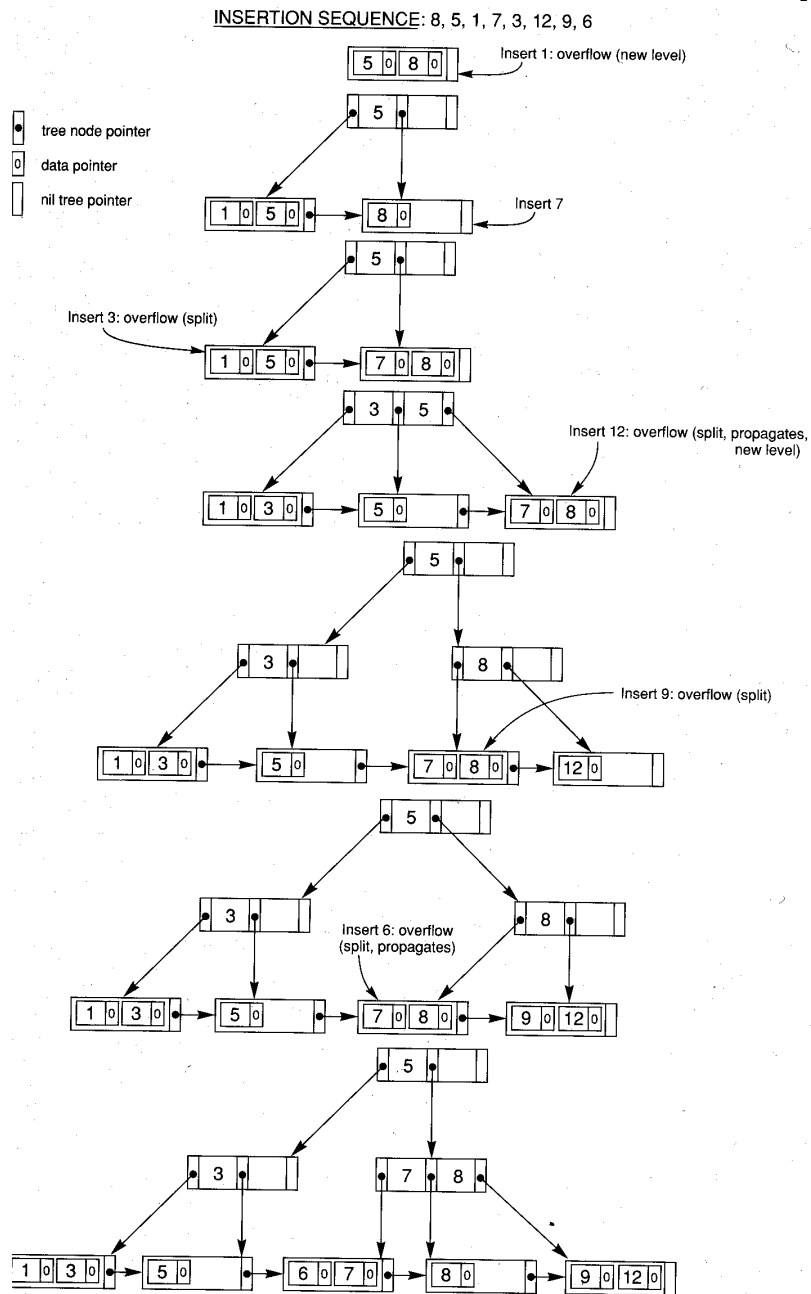


Figure 5.12 Example to illustrate insertion in a B<sup>+</sup>-tree

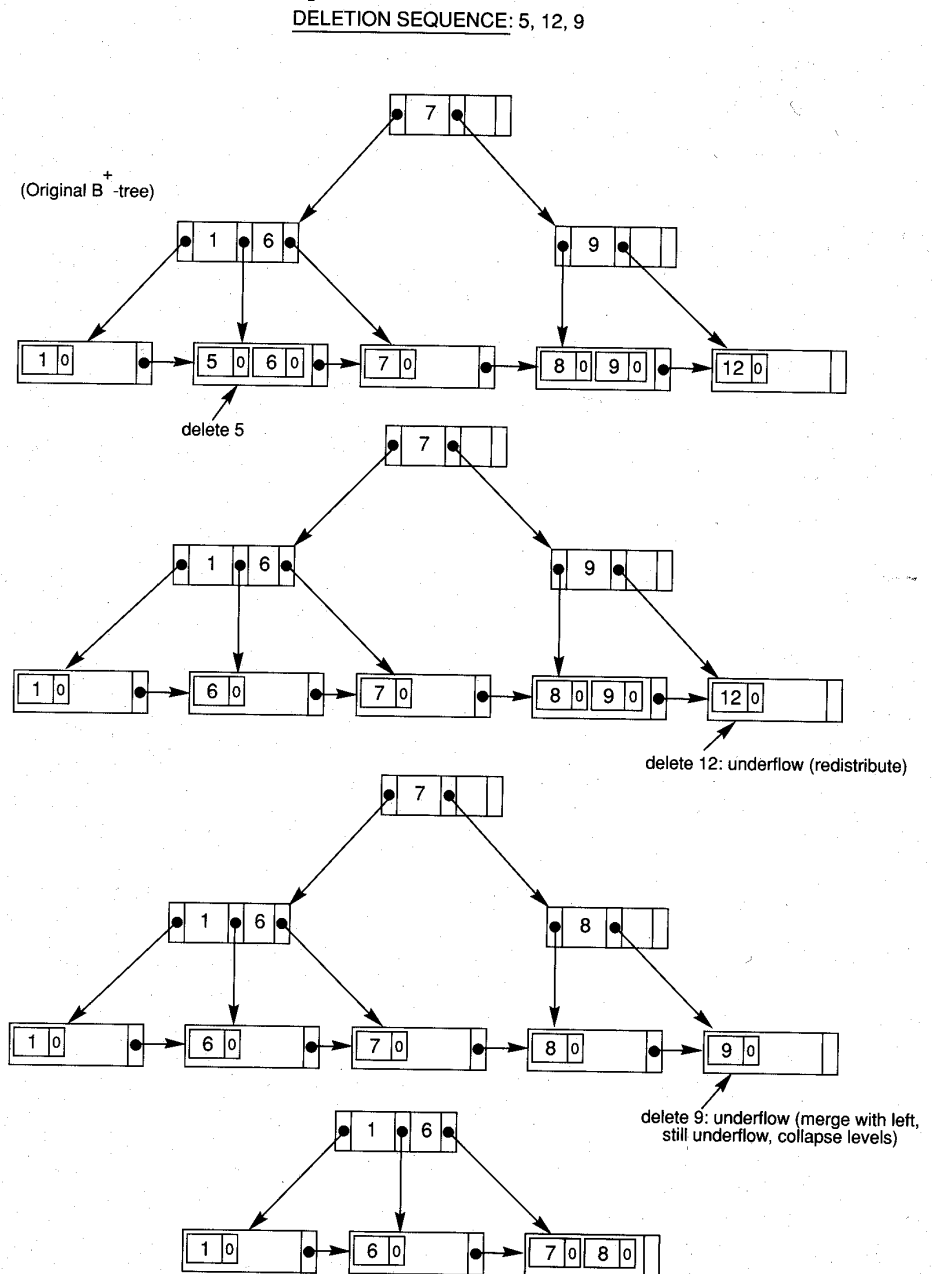
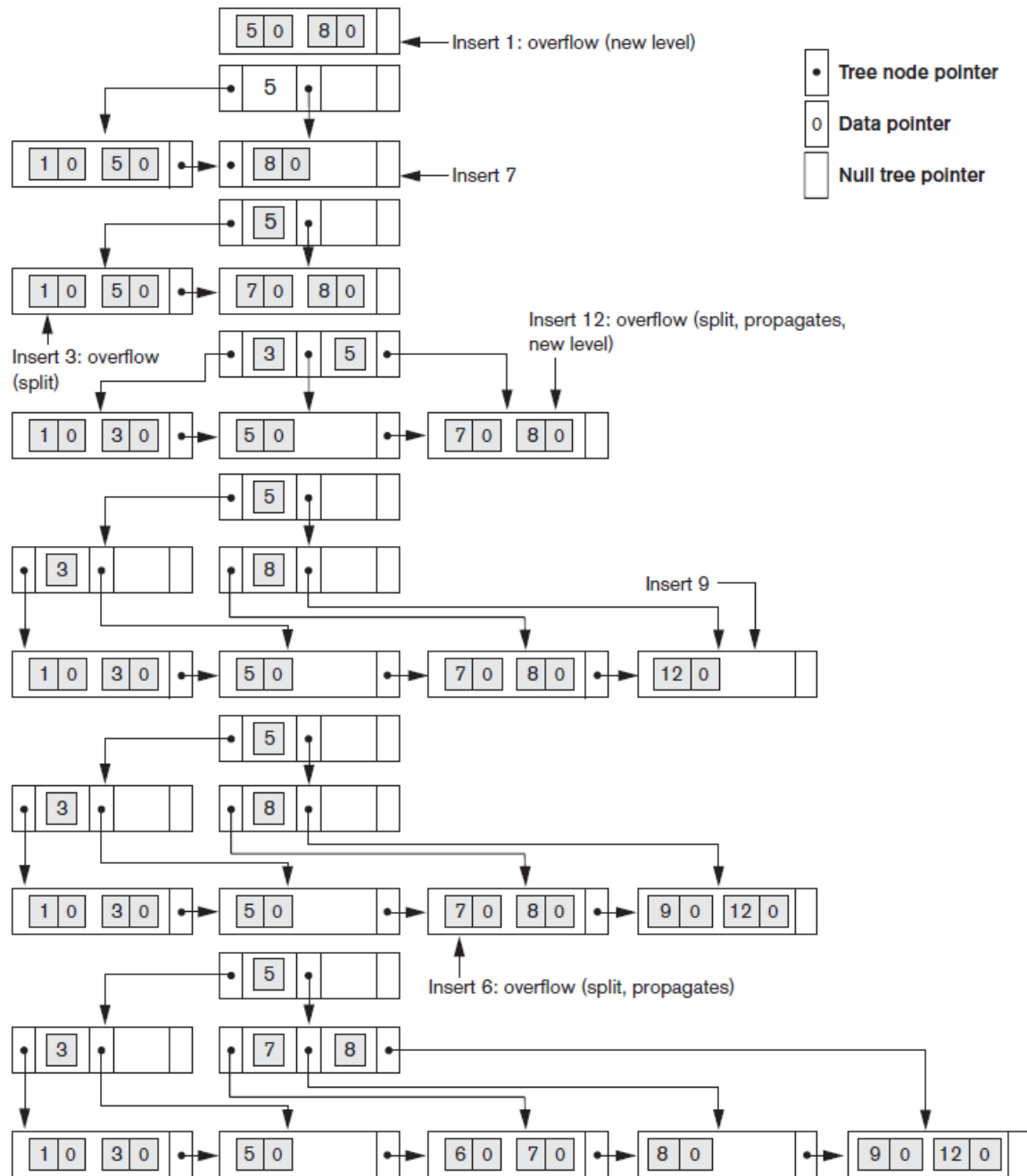
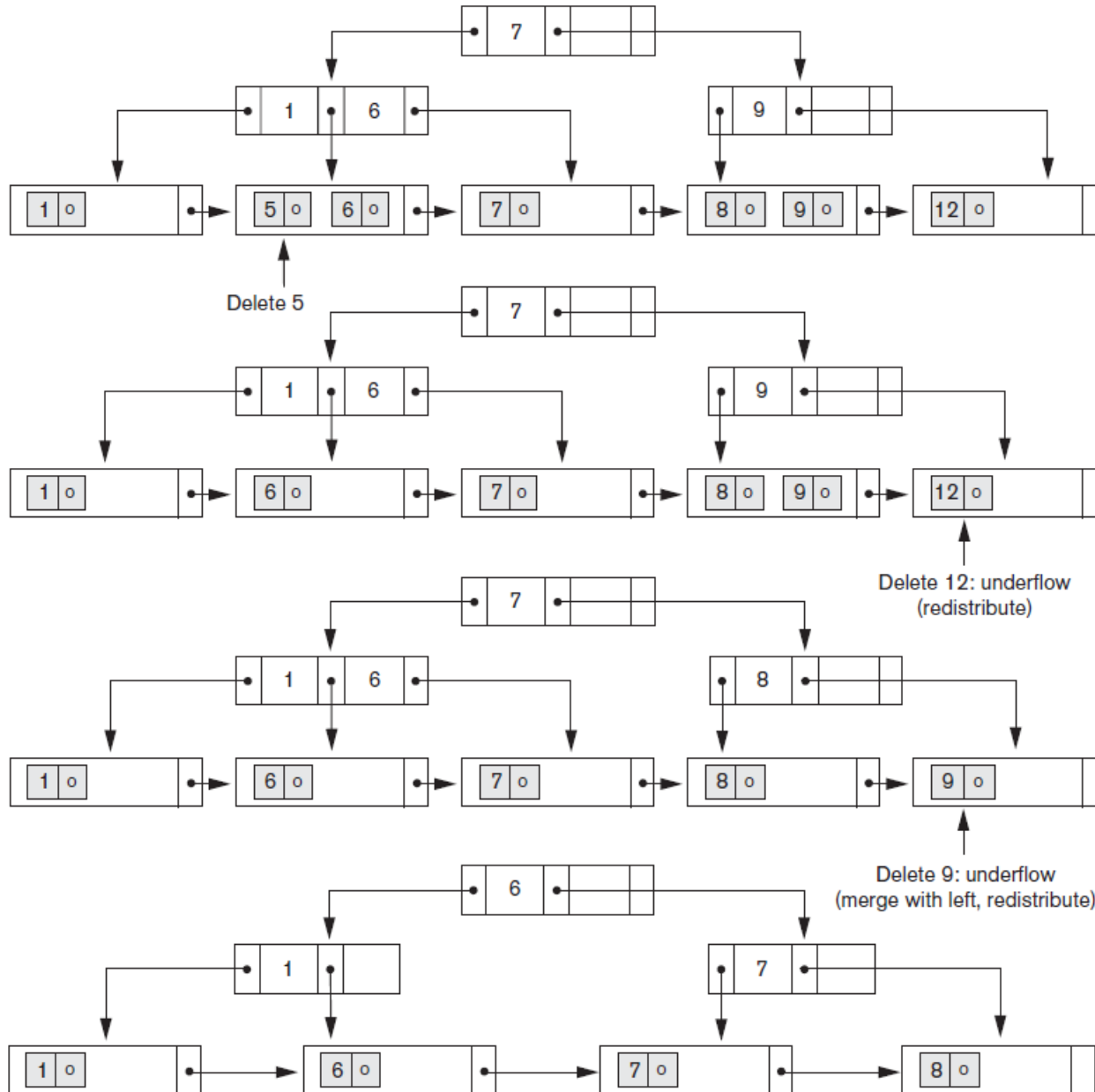


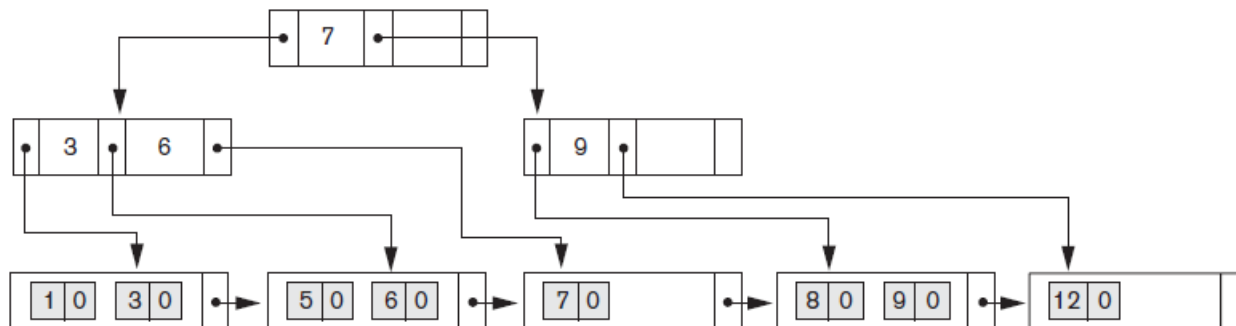
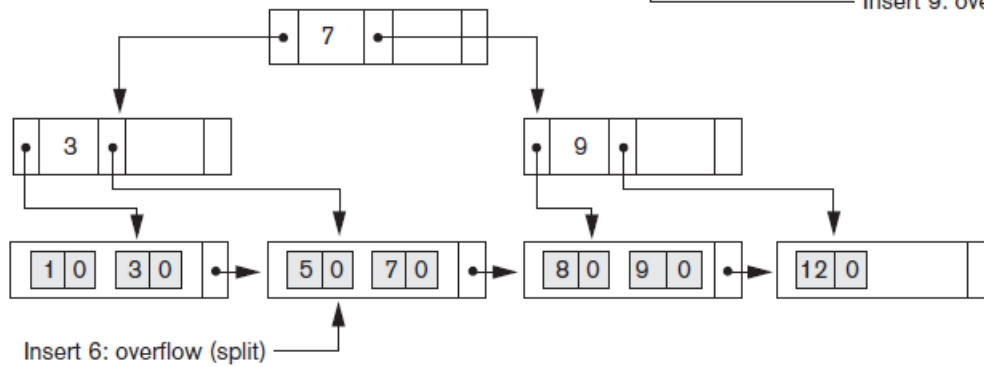
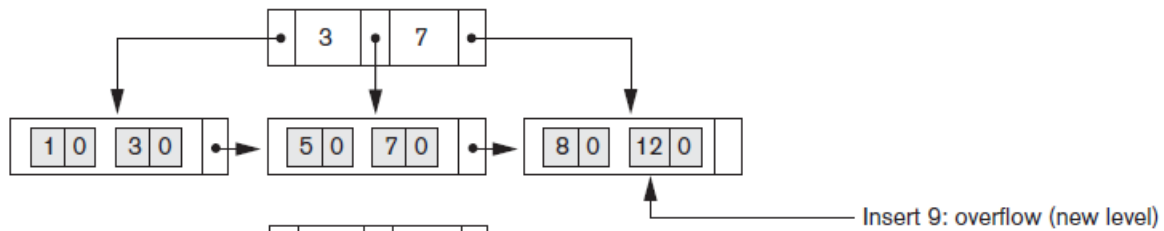
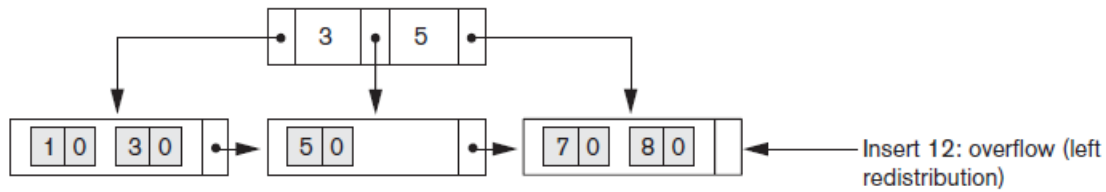
Figure 5.13 Example to illustrate deletion from a B<sup>+</sup>-tree

Insertion sequence: 8, 5, 1, 7, 3, 12, 9, 6



Deletion sequence: 5, 12, 9





# Zadaci

- Prikazati postupak formiranja B stabla reda 4 (tj.3 ključa po čvoru), ukoliko se ključevi dodaju sledećim redom: 12, 9, 7, 8, 3, 5, 6, 14, 13, 16, 15.
- Prikazati postupak brisanja ključeva sledećim redom: 5, 6.

