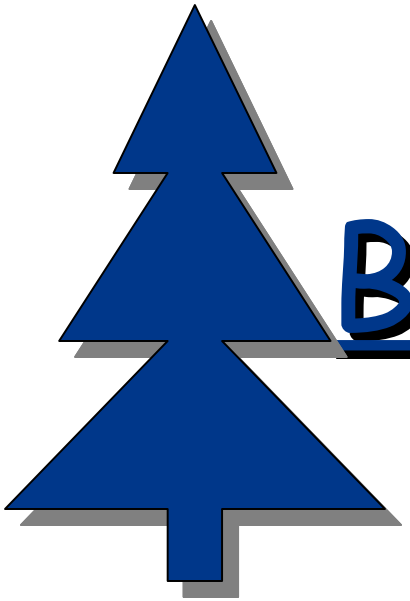
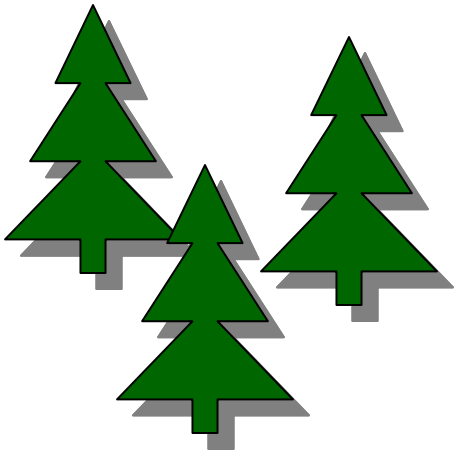


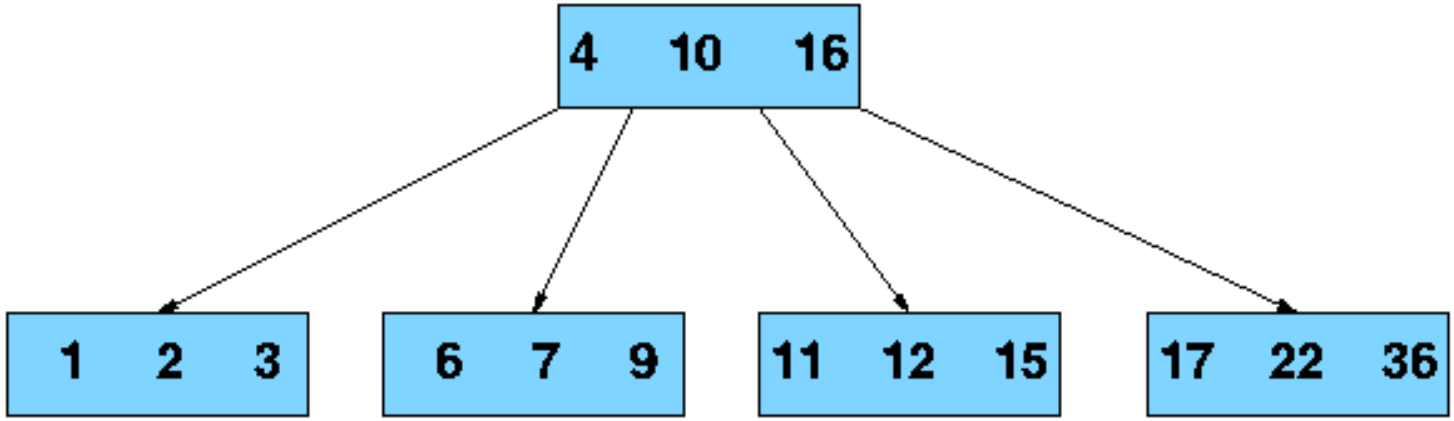
B-Strom



Definice B-stromu

- B-strom řádu *m* je strom, kde každý uzel má maximálně m následníků a ve kterém platí:
  1. Počet klíčů v každém vnitřním uzlu, je o jednu menší než je počet následníků (synů)
  2. Všechny listy jsou na stejné úrovni (mají stejnou hloubku)
  3. Všechny uzly kromě kořene mají nejméně $\left\lceil \frac{m}{2} \right\rceil$ následníků($\left\lceil \frac{m}{2} \right\rceil$ -1 klíčů)
  4. Kořen je buď to list, nebo má od 2 do m následníků
  5. Žádný uzel neobsahuje více než m následníků (m-1 klíčů)

Příklad B-Stromu



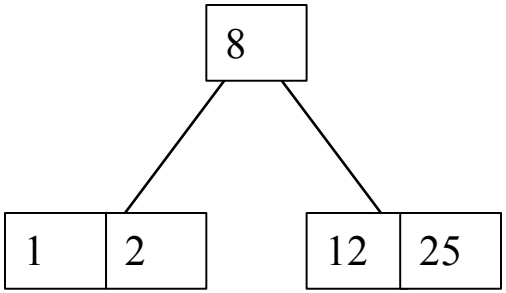
Vytvoření B-stromu

- Předpokládejme, že začínáme s prázdným B-stromem a ukládáme klíče v následujícím pořadí: **1 12 8 2 25 5 14 28 17 7 52 16 48 68 3 26 29 53 55 45**
- Chceme vytvořit B-strom řádu m=5, tj.každý uzel (kromě kořene) obsahuje nejméně 2 klíče a nejvýše 4 klíče.
- První 4 klíče :

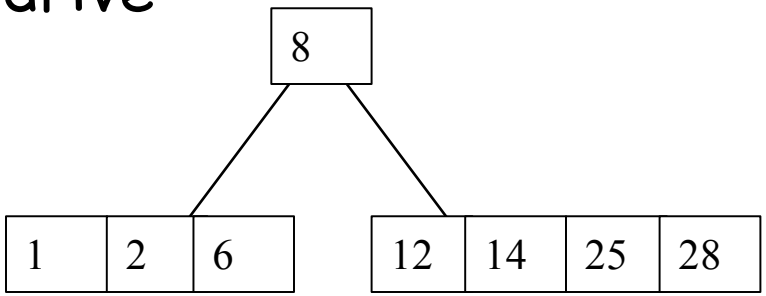
1	2	8	12

- Vložení páté položky (klíč 25) poruší podmínku 5 (dojde k tzv. přeplnění stránky)
- Přeplňená stránka se rozdělí na dvě, prostřední prvek se přesune do nadřazené stránky

Vytvoření B-stromu (pokračování)

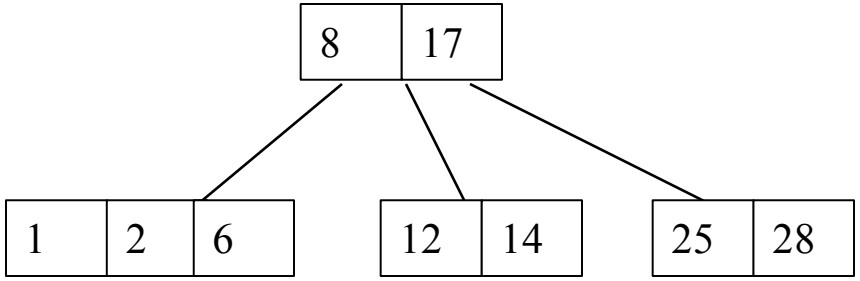


Další položky 6, 14, 28 budou vloženy do listů (listy se obsazují nejdříve

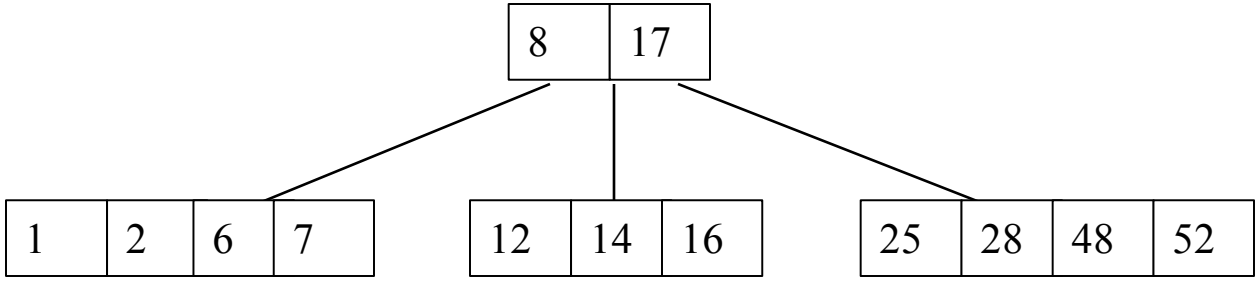


Vytváření B-stromu

Vložení 17 do pravé stránky způsobí přeplnění, stránka se rozdělí podle prostředního klíče a ten se přesune do kořene

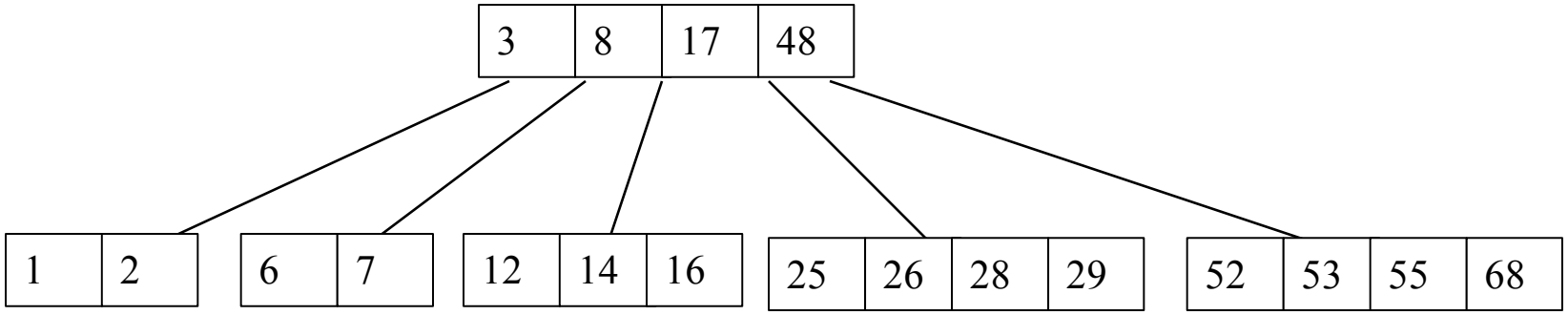


7, 52, 16, 48 se opět přidají do listů

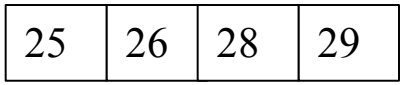


Vytváření B-stromu

Vložení 68 opět způsobí přeplnění stránky vpravo, klíč 48 se přesune do kořene, 3 přeplní levou stránku a po rozdělení přechází do kořene; 26, 29, 53, 55 jsou vloženy do listů

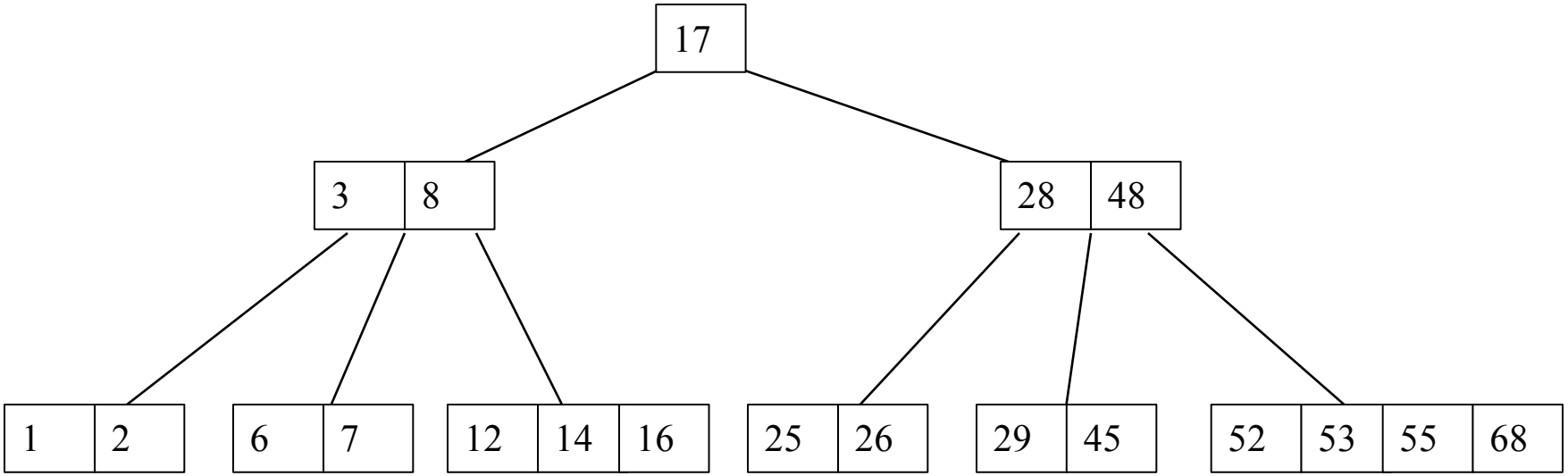


45 přeplní stránku



a klíč 28 se přesune do kořene, kde způsobí přeplnění a rozdělení kořenové stránky

Vytváření B-stromu

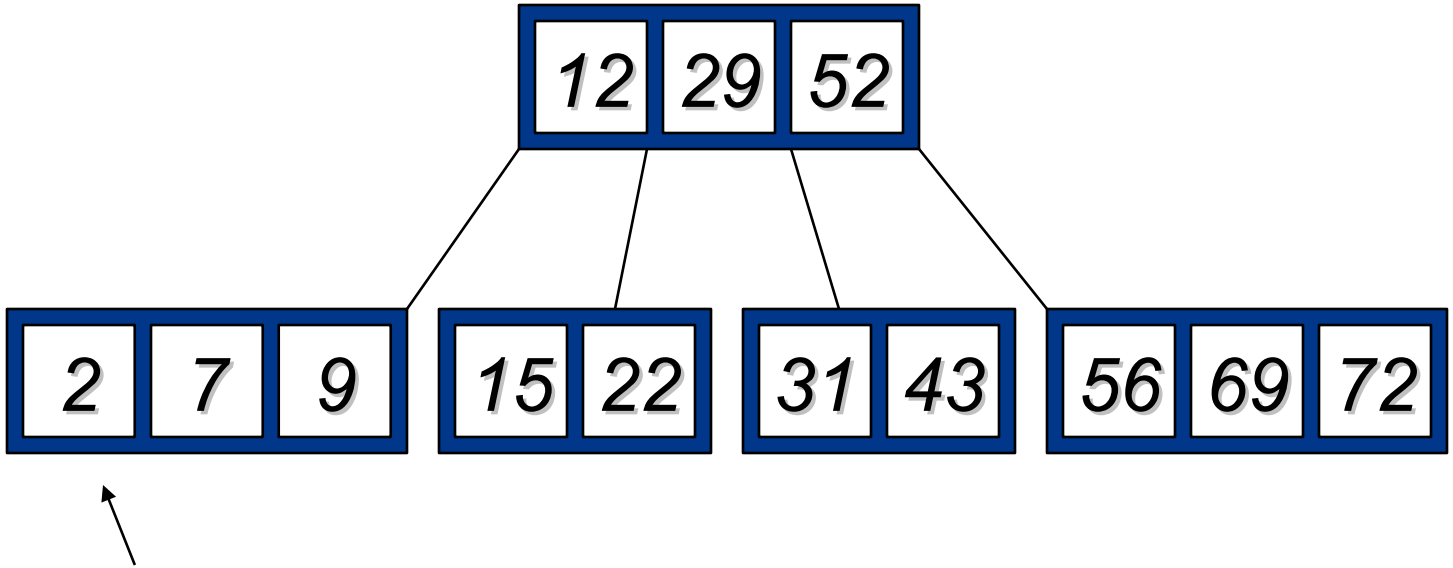


Vložení prvku do B-stromu (shrnutí)

- Nový prvek se vždy vkládá do listové stránky, ve stránce se klíče řadí podle velikosti.
- Pokud dojde k přeplnění listové stránky, stránka se rozdělí na dvě a prostřední klíč se přesune do nadřazené stránky (pokud nadřazená stránka neexistuje, tak se vytvoří)
- Pokud dojde k přeplnění nadřazené stránky předchozí postup se opakuje dokud nedojde k zařazení nebo k vytvoření nového kořene.

Rušení prvků B-stromu - rušení v listech bez podtečení velikosti stránky

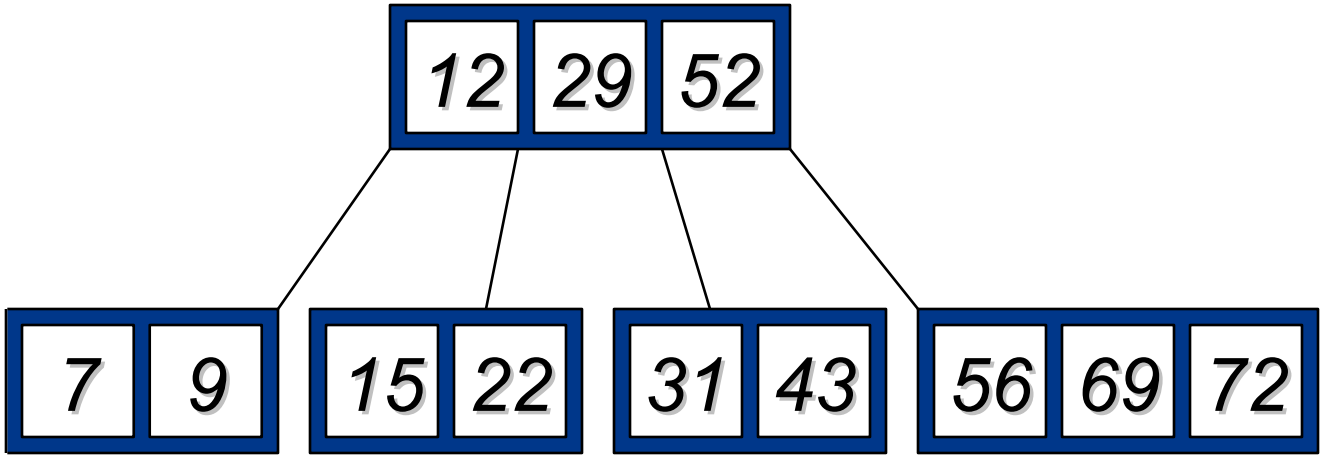
Předpokládejme B-strom 5 řádu...



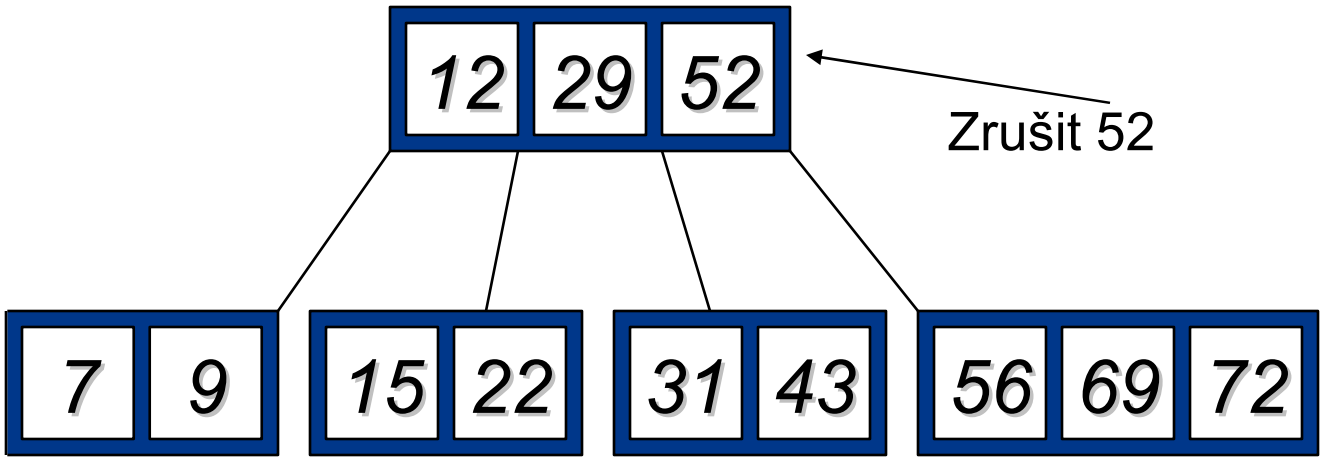
Zrušení klíče 2: Jelikož ve stránce je dostatečné množství klíčů,
Dojde pouze ke zrušení hodnoty 2 v listové stránce

Rušení prvků B-stromu - rušení v listech bez podtečení velikosti stránky

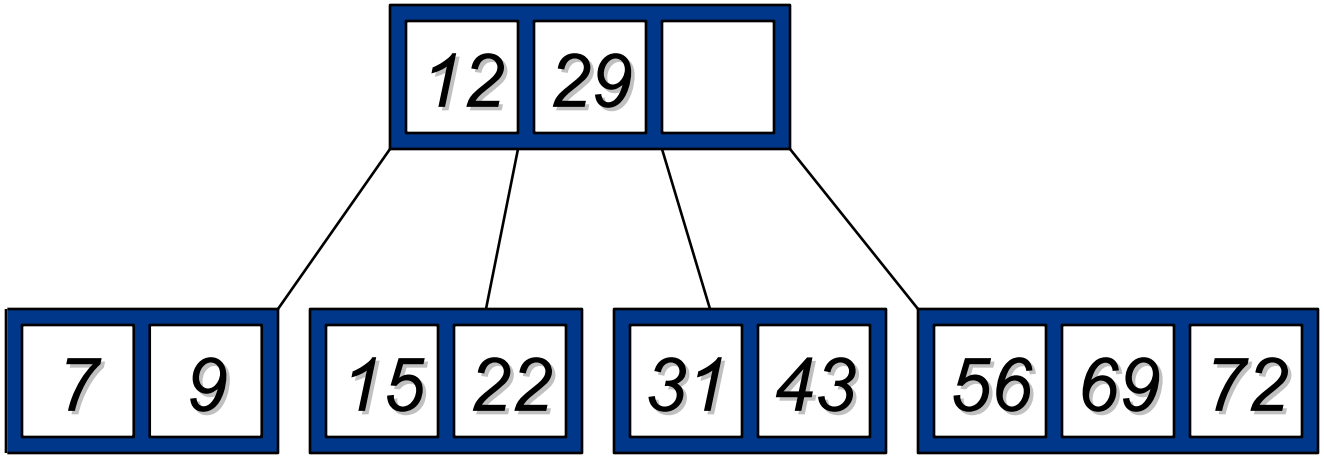
Předpokládejme B-strom 5 řádu...



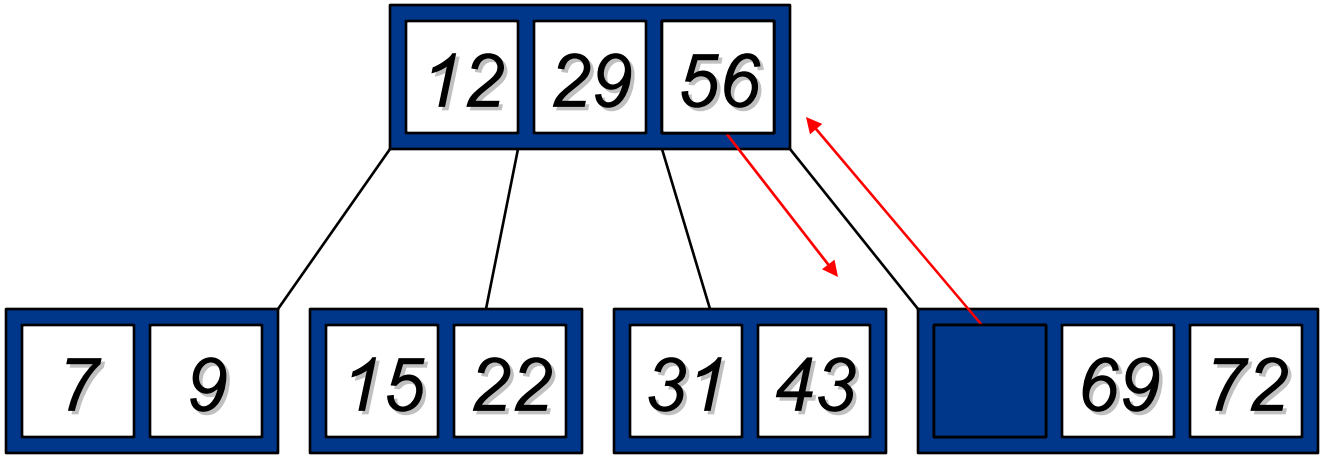
Rušení prvků B-stromu - rušení pvku v ostatních stránkách (kromě listů) bez podtečení velikosti stránky



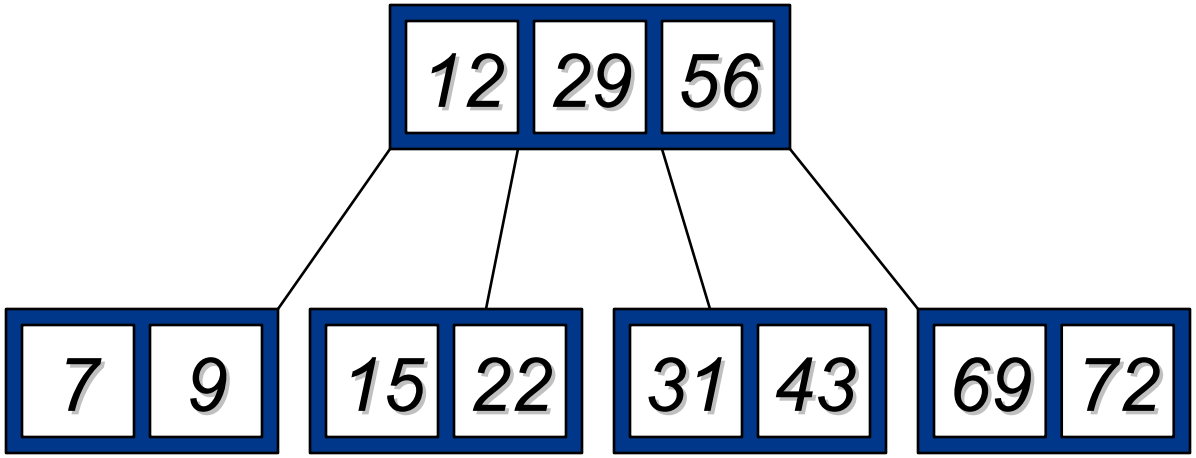
Rušení prvků B-stromu - rušení pvku v ostatních stránkách (kromě listů) bez podtečení velikosti stránky



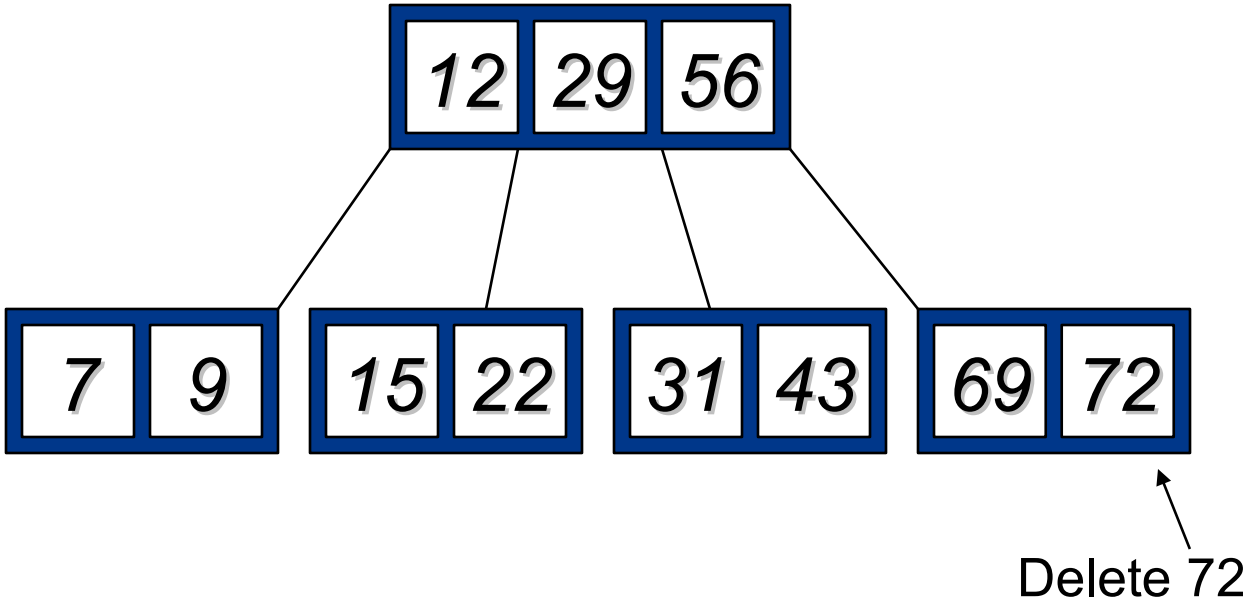
Rušení prvků B-stromu - rušení pvku v ostatních stránkách (kromě listů) bez podtečení velikosti stránky



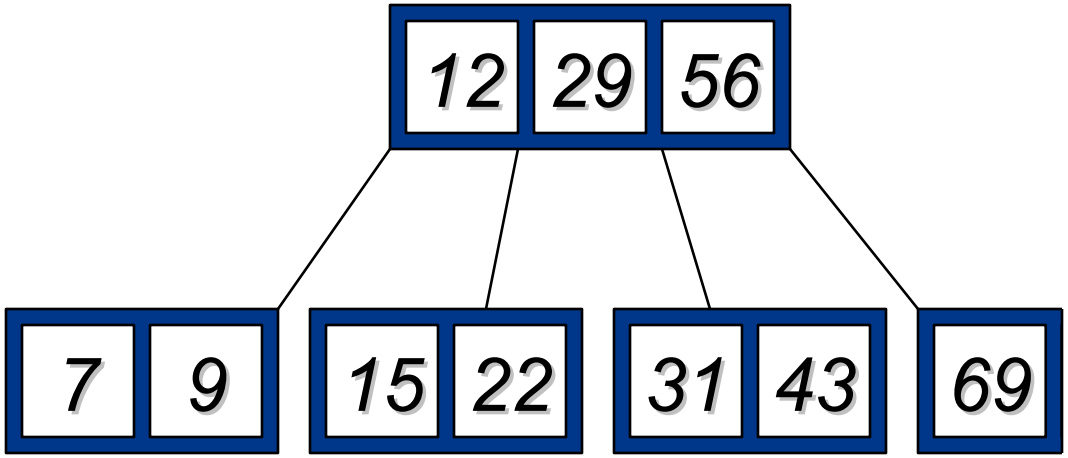
Rušení prvků B-stromu - rušení pvku v ostatních stránkách (kromě listů) bez podtečení velikosti stránky



Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují také minimální počet klíčů

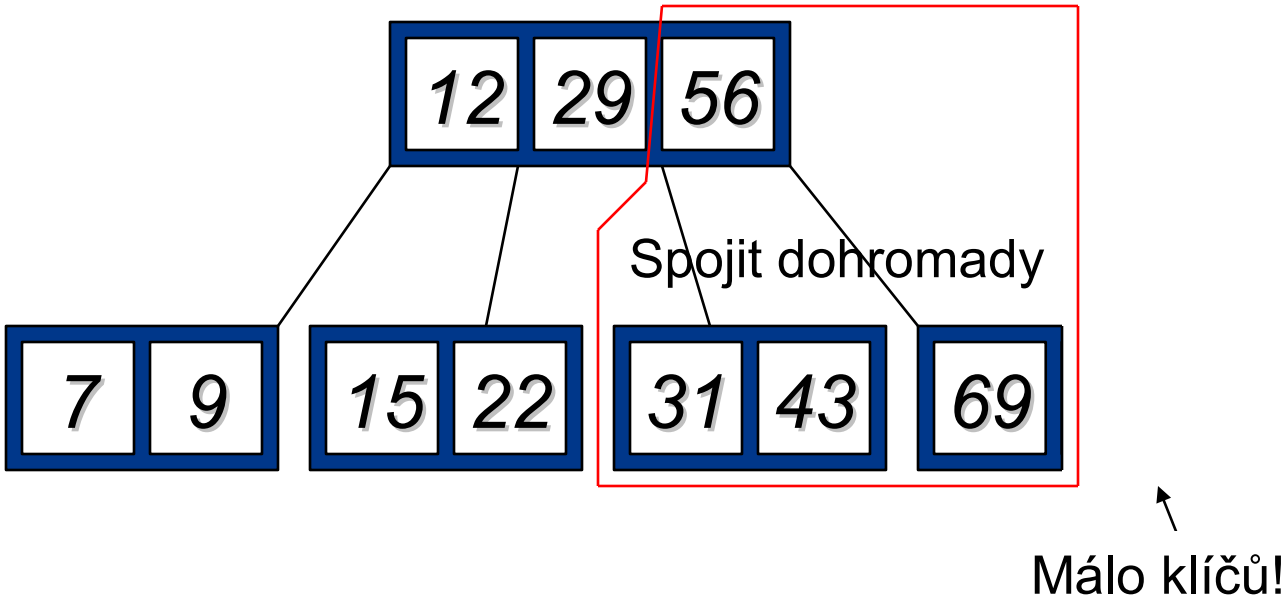


Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují také minimální počet klíčů

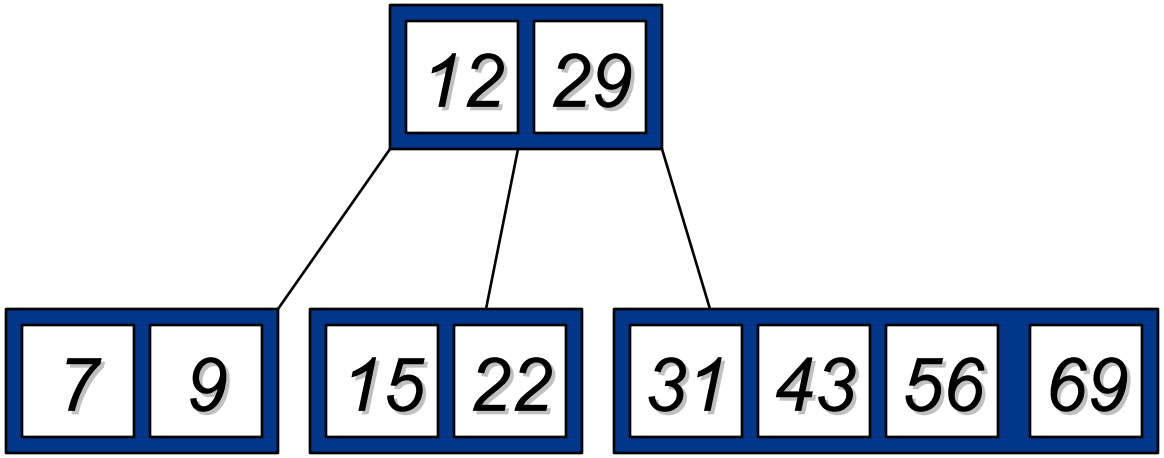


Málo klíčů!

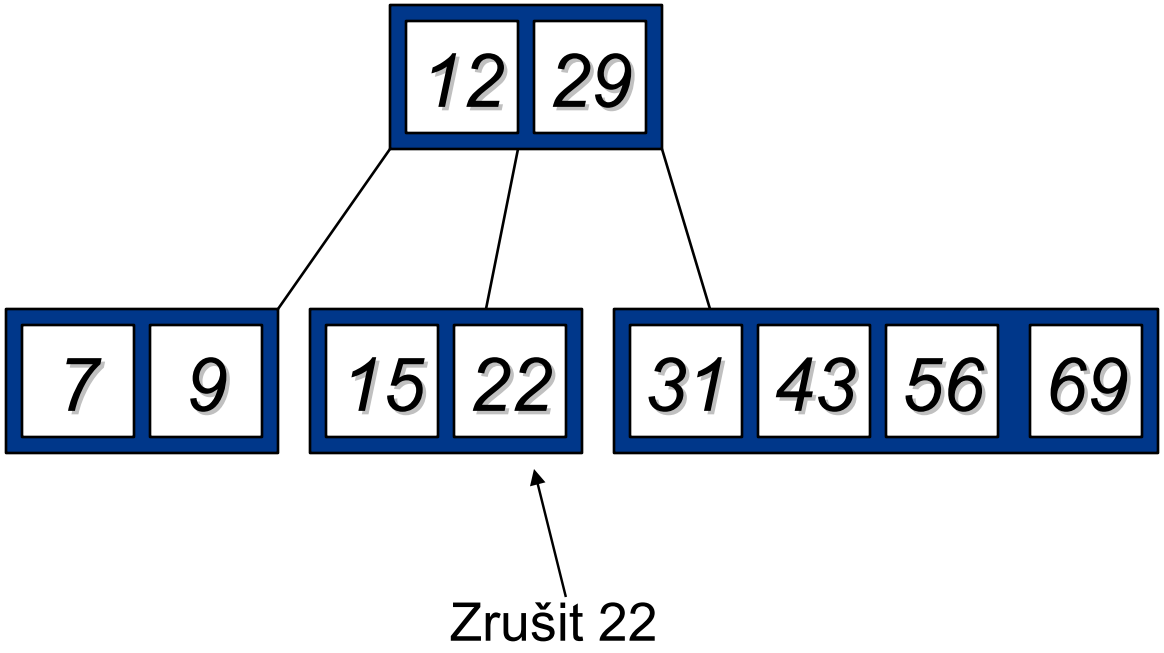
Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují také minimální počet klíčů



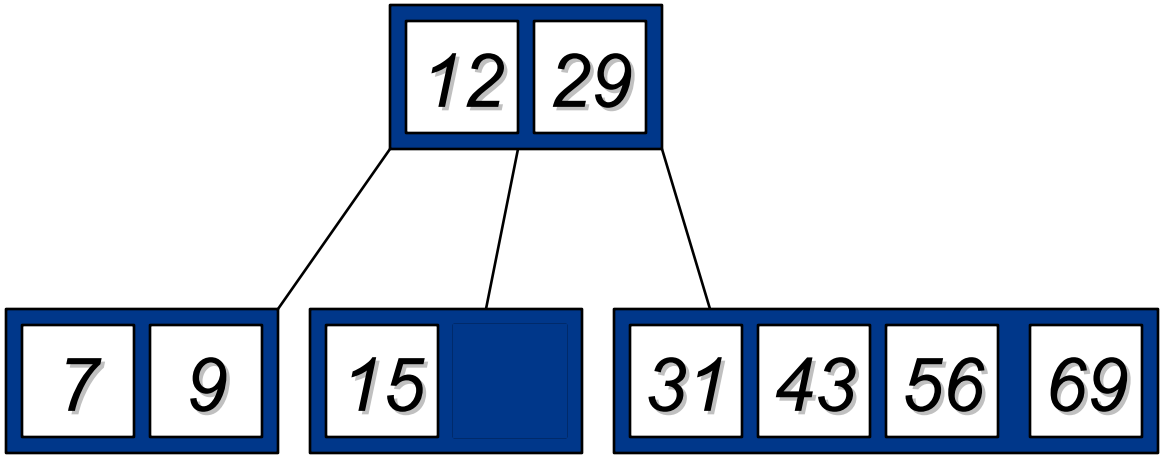
Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují také minimální počet klíčů



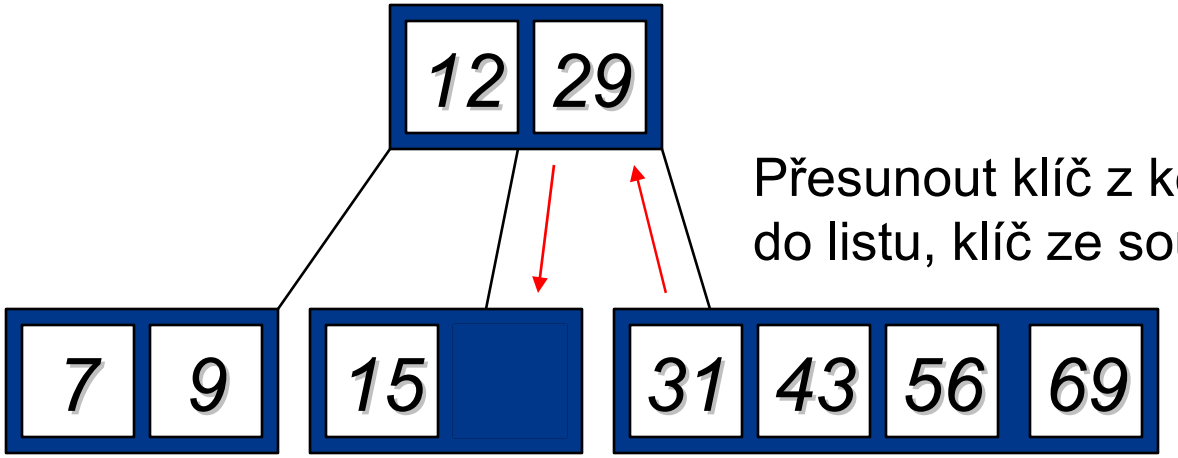
Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují dostatek klíčů



Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují dostatek klíčů

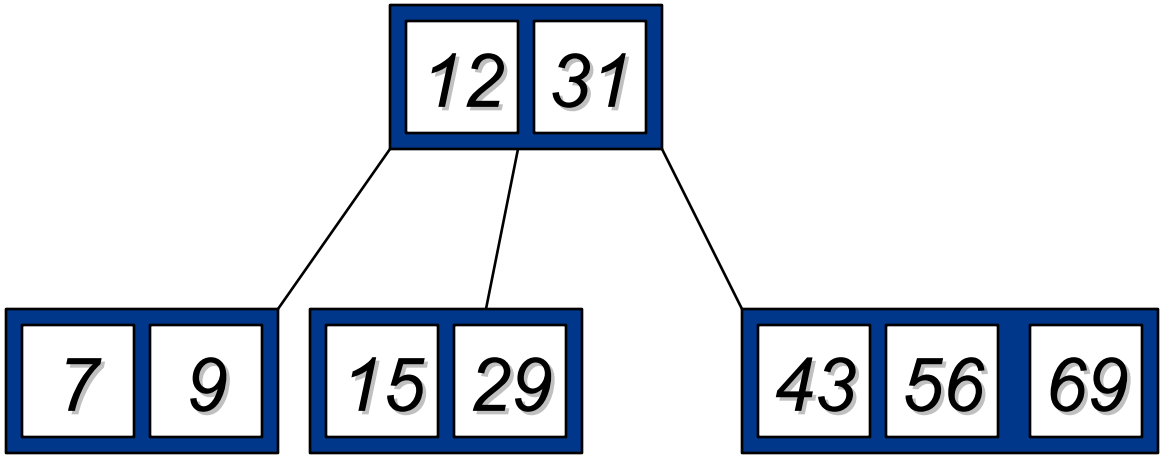


Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují dostatek klíčů



Přesunout klíč z kořene
do listu, klíč ze souseda do kořene

Rušení prvků ve stránce s minimálním počtem klíčů – sousední stránky obsahují dostatek klíčů



Analýza B-Stromů

- Maximální počet položek v B-stromu řádu m a výšky h:

kořen	$m - 1$
úroveň 1	$m(m - 1)$
úroveň 2	$m^2(m - 1)$
. . .	
úroveň h	$m^h(m - 1)$

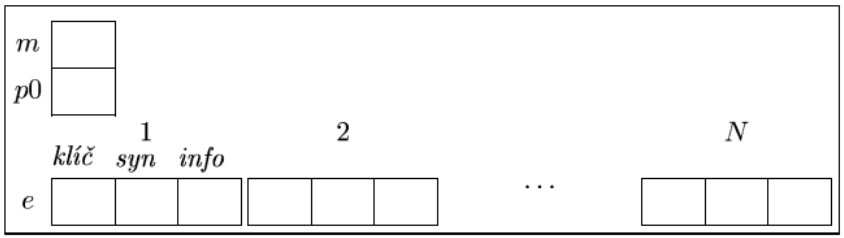
- Celkový počet položek je

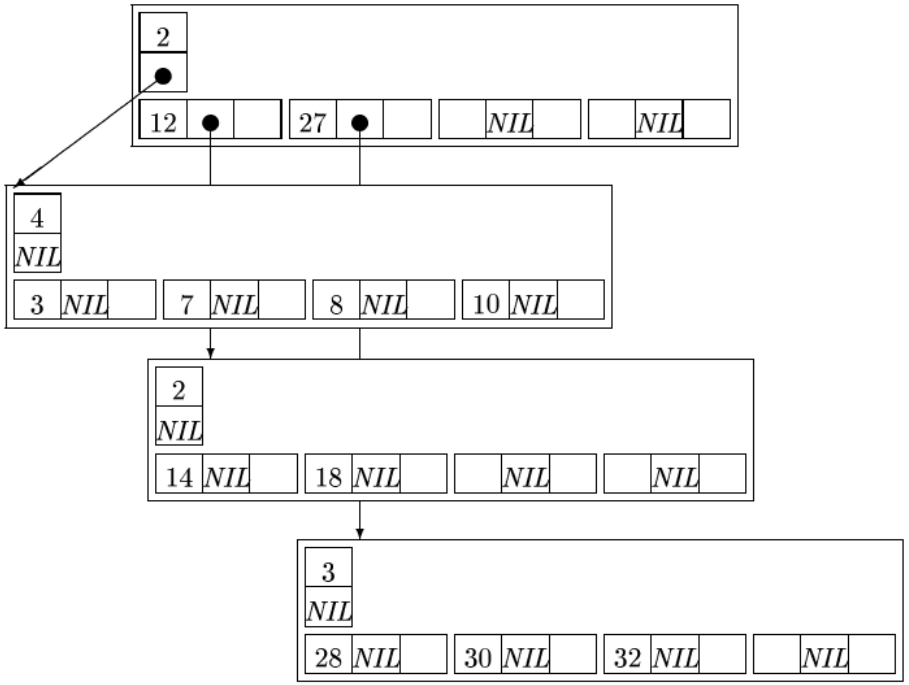
$$(1 + m + m^2 + m^3 + \ldots + m^h)(m - 1) =$$

$$[(m^{h+1} - 1)/ (m - 1)] (m - 1) = \mathbf{m^{h+1} - 1}$$

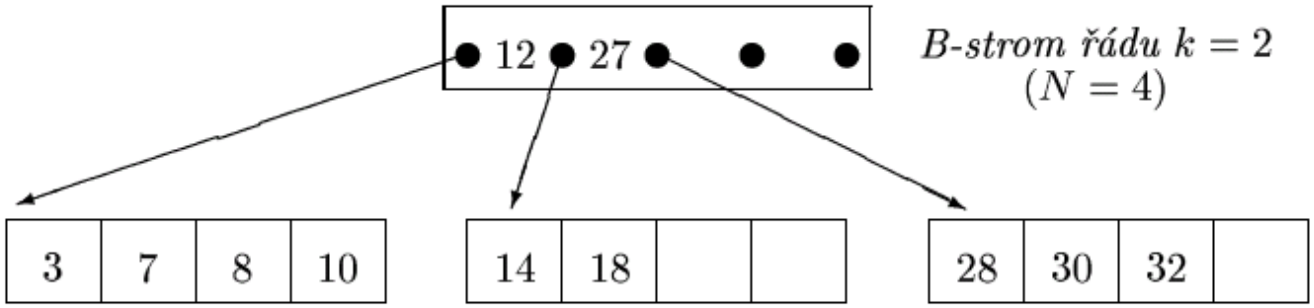
- Pokud $m = 5$ a $h = 2$ tak je celkem $5^3 - 1 = 124$ položek

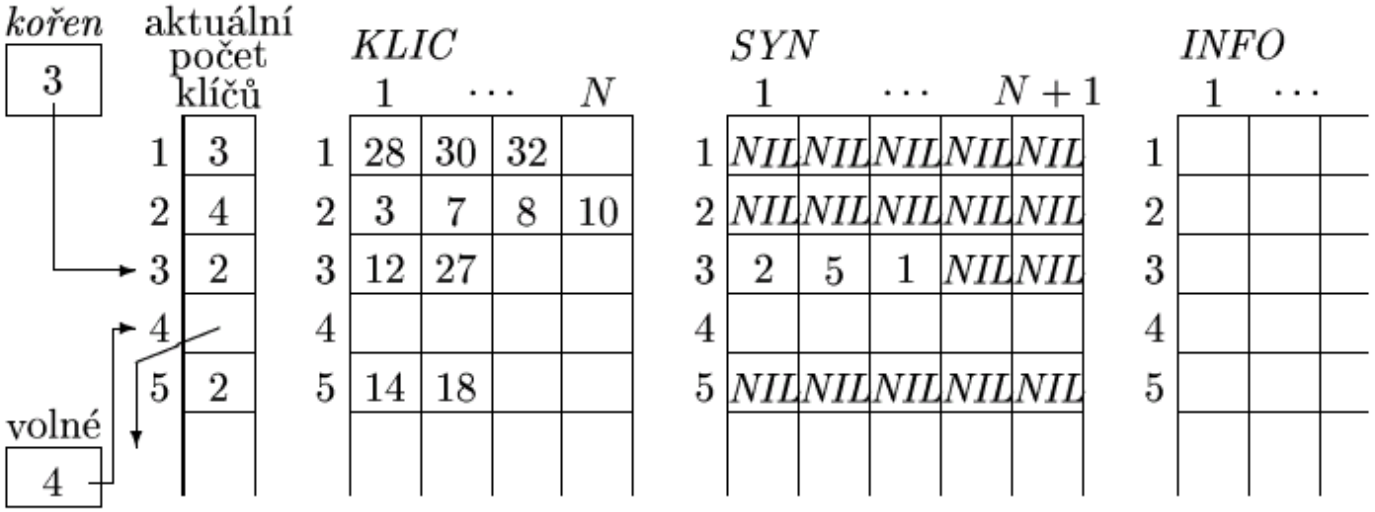
Implementace B-stromu - dynamická





Implementace B-stromu -statická





Vytvoření B-stromu

B-Tree-Create(T)

```
x <- Allocate-Node()
leaf[x] <- TRUE
n[x] <- 0
Disk-Write(x)
root[T] <- x
```

Vložení prvku do B-Stromu

B-Tree-Insert(T, k)

```
r <- root[T]
if n[r] = 2t - 1
      then s <- Allocate-Node()
           root[T] <- s
           leaf[s] <- FALSE
           n[s] <- 0
           c1 <- r
           B-Tree-Split-Child(s, 1, r)
           B-Tree-Insert-Nonfull(s, k)
      else B-Tree-Insert-Nonfull(r, k)
```

Vložení prvku do B-stromu

B-Tree-Insert-Nonfull(x, k)

```
i <- n[x]
if leaf[x]
      then while i >= 1 and k < key_i[x]
                do key_{i+1}[x] <- key_i[x]
                   i <- i - 1
           key_{i+1}[x] <- k
           n[x] <- n[x] + 1
           Disk-Write(x)
      else while i >= and k < key_i[x]
           do i <- i - 1
           i <- i + 1
           Disk-Read(c_i[x])
           if n[c_i[x]] = 2t - 1
                then B-Tree-Split-Child(x, i, c_i[x])
                     if k > key_i[x]
                          then i <- i + 1
           B-Tree-Insert-Nonfull(c_i[x], k)
```

Rozdělení stránky

B-Tree-Split-Child(x, i, y)

```
z <- Allocate-Node()
leaf[z] <- leaf[y]
n[z] <- t - 1
for j <- 1 to t - 1
     do key_j[z] <- key_{j+t}[y]
if not leaf[y]
     then for j <- 1 to t
          do c_j[z] <- c_{j+t}[y]
n[y] <- t - 1
for j <- n[x] + 1 downto i + 1
     do c_{j+1}[x] <- c_j[x]
c_{i+1} <- z
for j <- n[x] downto i
     do key_{j+1}[x] <- key_j[x]
key_i[x] <- key_t[y]
n[x] <- n[x] + 1
Disk-Write(y)
Disk-Write(z)
Disk-Write(x)
```

Vyhledávání v B-Stromu

B-Tree-Search(x, k)

```
i <- 1
while i <= n[x] and k > key_i[x]
      do i <- i + 1
if i <= n[x] and k = key_i[x]
      then return (x, i)
if leaf[x]
      then return NIL
      else Disk-Read(c_i[x])
           return B-Tree-Search(c_i[x], k)
```