Podatkovne Strukture in Algoritmi

Vaje

4.10.24

Vaje 1

$$O(f(n)) = \{g(n) : (\exists c > 0)(\exists n_0 \in \mathbb{N})(\forall n)(n \geq n_0 \Rightarrow g(n) \leq c \cdot f(n)\}$$

$$o(f(n)) = \{g(n) : (\forall c > 0)(\exists n_0 \in \mathbb{N})(\forall n)(n \geq n_0 \Rightarrow g(n) < c \cdot f(n)\}$$

$$\Omega(f(n)) = \{g(n) : (\exists c > 0)(\exists n_0 \in \mathbb{N})(\forall n)(n \geq n_0 \Rightarrow g(n) \geq c \cdot f(n)\}$$

$g = \Theta(f(n)) \Leftrightarrow g = O(f(n)) \wedge g = \Omega(f(n))$

Velja: $g(n) = o(f(n)) \Leftrightarrow \lim_{n \to \infty} \frac{g(n)}{f(n)} = 0$

Naloga 1

Algoritem ima časovno zahtevnost $T(n) = \frac{1}{2}n^2 - 3n$.
Pokaži, da je $T(n) = \Theta(n^2)$

a) $T(n) = O(n^2)$

$g(x)$ $f(x)$

Iščemo $c > 0$ in $n_0 \in \mathbb{N}$, da velja
$\frac{1}{2}n^2 - 3n \leq c \cdot n^2$ za $n \geq n_0$.

$\frac{1}{2} - \frac{3}{n} \leq c$

Vzamemo $c = \frac{1}{2}$. Torej $\frac{1}{2} - \frac{3}{n} < \frac{1}{2}$ $\frac{3}{n} > 0$

Vzamemo $n_0 = 1$.

b) $T(n) = \Omega(n^2)$

Iščemo $c > 0$ in $n_0 \in \mathbb{N}$, da za vse $n \geq n_0$ velja

$$\frac{1}{2}n^2 - 3n \geq cn^2$$
$$\frac{1}{2} - \frac{3}{n} \geq c$$

Vzamemo $c = \frac{1}{4}$.

$$\frac{1}{2} - \frac{3}{n} \geq \frac{1}{4} \quad / 4n$$
$$2n - 12 \geq 1 \Rightarrow n \geq 12 \Rightarrow \text{Vzamemo } n_0 = 12$$

Naloga 2

Pokaži, da velja:

a) $2^{n+1} = O(2^n)$

b) $2^{2n} \neq O(2^n)$

a) Iščemo $c > 0$ in $n_0 \in \mathbb{N}$, da za vse $n \geq n_0$ velja:

$$2^{n+1} \leq c2^n$$
$$2 \cdot 2^n \leq c \cdot 2^n$$
$$2 \leq c$$

Vzamemo $c = 2$ in $n_0 = 1$, ker je neodvisno od n.

Opomba: če je $f = O(g)$ potem je $g \neq O(f)$

b) 1. način: po def.

$\leftarrow$ $g \neq O(f(n)) \Leftrightarrow (\forall c > 0)(\forall n_0 \in \mathbb{N})(\exists n)((n \geq n_0) \wedge g(n) > cf(n))$

Naj bosta $c > 0$ in $n_0 \in \mathbb{N}$ poljubna. Iščemo n, da velja $n \geq n_0$ in $2^{2n} > c \cdot 2^n$. Sledi

$$2^{2n} > c2^n \quad / : 2^n$$
$$2^n > c \quad / \log_2$$
$$n > \log_2 c$$

Vzamemo $n = \log_2 c + n_0$ $(n(c, n_0))$

2. način: z limito

$$\lim_{n\to\infty} \frac{2^n}{2^{2n}}$$
$$= \lim_{n\to\infty} \frac{2^n}{2^n \cdot 2^n}$$
$$= \lim_{n\to\infty} \frac{1}{2^n} = \underline{\underline{0}}$$

Naloga 3

Algoritem z izbiro

Vhodni podatki: polje A celih števil z dolžino n.

Izhodni podatki: naraščajoče urejeno polje A

```
for i=1 to n-1 do
    minIndex= i
    for j=i+1 to n
        if A[j] < A[minIndex] then
            minIndex =j
    swap (A[i],A[minIndex])
```

a) Pokaži pravilnost delovanja algoritma z izbiro uporabljajoč zančno invarianco

b) Analiziraj njegovo časovno zahtevnost.

a) Zančna invarianca: na začetku ite iteracije velja, da je podpolje A[1,...,i-1] urejeno in vsebuje i-1 najmanjših elementov v A.

Baza: i=1. Na začetku 1. iteracije velja, da je podpolje A[1,...,0] urejeno in vsebuje 0 najmanjših elementov v A.

Ind. korak: $i \mapsto i+1$. Pokažimo, da na začetku $(i+1)$ iteracije velja, da je podpolje $A[1,\dots,i]$ urejeno in vsebuje prvih i najmanjših elementov v A.

Dovolj je pokazati, da je na začetku $(i+1)$ iteracije element $A[i]$ i-ti najmanjši v A. Poglejmo si i-to iteracijo. Notranja zanka poišče i-ti najmanjši element v A, ki je shranjen v $A[\text{min\_indeks}]$. Klic swap$[A[i], A[\text{min_index}]]$ zagotovi, da je i-ti najmanjši element v $A[i]$.

Poglejmo: $i = n$. Po zanki invarianci je podpolje $A[1,\dots,n-1]$ urejeno in vsebuje $(n-1)$ prvih najmanjših elementov. Očitno mora biti A urejeno.

11. 10. 24

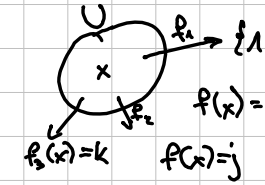
Vaje 2

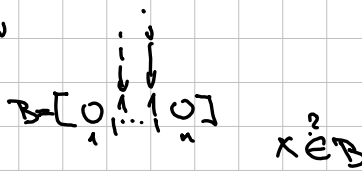
Programerske DN: navodila v pdf (DN 0 ni obvezna, priporočljiva)

Na učilnici

Bloomov filter

$B = [\underset{1}{0}, \dots, \underset{n}{0}]$ → polje samih ničel





Vaje 1, nal. 3 do konca

```
for i = 1 to n-1 do
    minIndex = i
    for j = i+1 to n
        if A[j] < A[minIndex] then
            minIndex = j
    swap(A[i], A[minIndex])
```

Čas. zahtevnost
$c_1 \cdot n$
$c_2 \cdot (n-1)$
$c_3 \cdot \sum_{i=1}^{n-1} (n-i+1)$
$c_4 \cdot \sum_{i=1}^{n-1} (n-i)$
$c_5 \sum_{i=1}^{n-1} t_i$
$c_6 \cdot (n-1)$

t_i ... št., bi pove, kolikokrat je if stavek izpolnjen pri danem i.

Splošno:

$$T(n) = c_1 n + c_2(n-1) + c_3 \sum_{i=1}^{n-1} (n-i+1) + c_4 \sum_{i=1}^{n-1} (n-i) + c_5 \sum_{i=1}^{n-1} t_i + c_6(n-1)$$

Best case ($t_i = 0, \forall i \Leftrightarrow$ A je že urejen)

$$T(n) = c_1 n + c_2(n-1) + c_3 \sum_{i=1}^{n-1} (n-i+1) + c_4 \sum_{i=1}^{n-1} (n-i) + c_6(n-1)$$

- DN: Poračunaj vsote (podoben worst case-u)

Worst case ($t_i = n-i, \forall i \Leftrightarrow$ A je urejen v obratnem vstrem redu)

$$T(n) = c_1 n + c_2(n-1) + c_3 \sum_{i=1}^{n-1} (n-i+1) + c_3 \underbrace{\sum_{i=1}^{n-1} 1}_{(n-1)} + c_4 \sum_{i=1}^{n-1} (n-i) + c_5 \sum_{i=1}^{n-1} (n-i) + c_6(n-1)$$

$$= (-c_2 - c_3 - c_6) + (c_1 + c_2 + c_3 + c_6)\, n + (c_3 + c_4 + c_5) \sum_{i=1}^{n-1} (n-1)$$

Vsota 1. m naravnih št.:

$$\sum_{i=1}^{m} i = 1 + 2 + 3 + \ldots + m = \frac{m(m+1)}{2}$$

$$= (-c_2 - c_3 - c_6) + (c_1 + c_2 + c_3 + c_6) n + (c_3 + c_4 + c_5) \frac{(n-1) \cdot n}{2}$$

$$= (-c_2 - c_3 - c_6) + (c_1 + c_2 + \tfrac{1}{2} c_3 - \tfrac{1}{2} c_4 - \tfrac{1}{2} c_5 - \tfrac{1}{2} c_6) n + \frac{(c_3 + c_4 + c_5)}{2} n^2$$

$$= A + Bn + Cn^2 = \Theta(n^2)$$

1. Imamo naslednjo kodo:

```
int FooBar(A)
    n = len(A); B = newarray(n)
    for j:=0 to n-1 {
        B[j] = 0
        for i:= j to 0 step -1
            if A[i] < A[j]{
                B[j] = B[j]+1
            }
        }
    }
    return B
```

a) Naj bo A[56, 47, 66, 71, 17, 19, 92]. Kaj vrne FooBar?
b) Kaj vsebuje vektor B, ki ga vrne FooBar(A) v splošnem?
c) Poišči zančno invarianco za notranjo for zanko in jo pokaži.
d) Analiziraj čas. zahtevnost alg. (DN)

c) = dokaz za a) in b)

a) n = 7

	0	1	2	3	4	5	6
B =	0	0	∅	∅	0	∅	∅
			1	1		1	6

j = 0: i = 0 // 2 2
j = 1: i = 1 // 3
i = 0 //
j = 2: i = 2 //
i = 1 +1
i = 0 +1
j = 3: i = 3 //
i = 2 +1
i = 1 +1
i = 0 +1

B = [0, 0, 2, 3, 0, 1, 6]

b) B[j] ... št. ele. v A[0, ..., j], ki je manjših od A[j]

c) 2.1.: Pri danem j, po vsaki iteraciji i velja, da je v B[j] shranjeno št. ele. v A[i,...j], ki so manjši od A[j]

Dokaz z indukcijo:

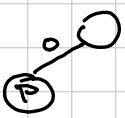
BI: i=j: št. ele. v A[j,...j] je 1 in št. manjših od A[j] je 0, kar velja ker je B[j]=0 inicializiran

IK: i→i-1: Pokazati moramo, da po iteraciji (i-1) velja, da je v B[j] shranjeno št. ele. v A[i-1,...,j], ki so manjši od A[j]. Po I.P., po iteraciji i velja, da je v B[j] shranjeno št. ele. v A[i,...,j], ki so manjši od A[i]. Poglejmo si iteracijo (i-1). V tej iteraciji primerjamo A[i-1] z A[j]. Če je A[i-1]<A[j], vrednosti B[j] prištejemo 1 in rezultat sledi.

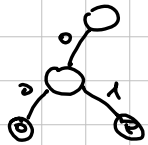
Vaje 3

1. Imamo dvojiško drevo $G = \{0,1\}$ in naslednje elemente:
(01010011, P), (00000111, o), (00100001, d), (01010001, a),
(11101100, t), (10010101, e), (01001010, k).
a) Ustavi zg. $\omega(\cdot)$ v številsko drevo.
b) Nariši pripadajoče Patricijevo drevo.
c) Patricijevo drevo stisni po prvih dveh plasteh, prvih treh plasteh, prvih štirih plasteh in prvih petih plasteh. Za katero stiskanje bi se odločili in zakaj?

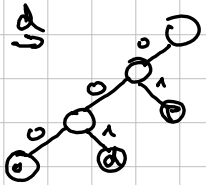
a) P



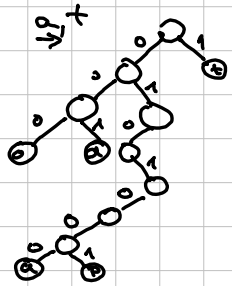
o



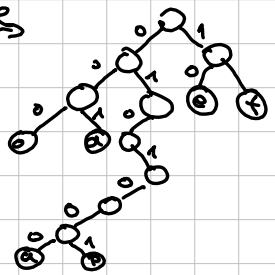
d



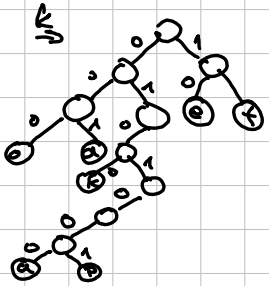
t



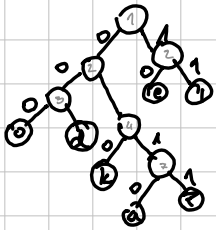
e



k

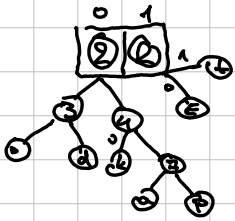


b)

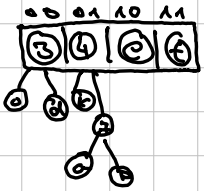


Čas. zahtevnost ostane enaka
Pros. zahtevnost se zmanjša

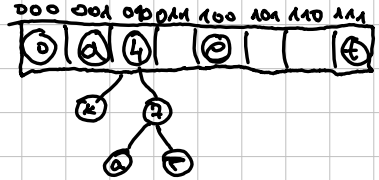
c) Po 2 plasteh



Po 3 plasteh



Po 4 plasteh



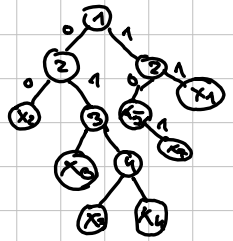
Po 5 plasteh (DN

2. Imamo naslednje ključe nad abecedo $\Sigma = \{0,1\}$:

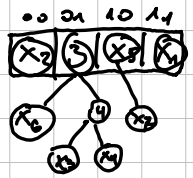
$x_1 = 111110$
$x_2 = 00010101110$
$x_3 = 01100000$
$x_4 = 011110$
$x_5 = 101$
$x_6 = 01011$
$x_7 = 10111111$

a) Ustavi dane ključe v Patricijevo drevo.
b) Patricijevo drevo stisnite po plasteh za $\alpha = \frac{3}{4}$
c) Recimo, da imamo v št. drevesu n ključev. Kakšne naj bodo njihove vrednosti, da po stiskanju po plasteh $\alpha = \frac{3}{4}$ ne bo nobenega polja? D.r., za vsako stiskanje po plasteh bi bila gostota vedno premajhna

a) $x_2 = 00010101110$
$x_6 = 01011$
$x_3 = 01100000$
$x_4 = 011110$
$x_5 = 101$
$x_7 = 10111111$
$x_1 = 111110$

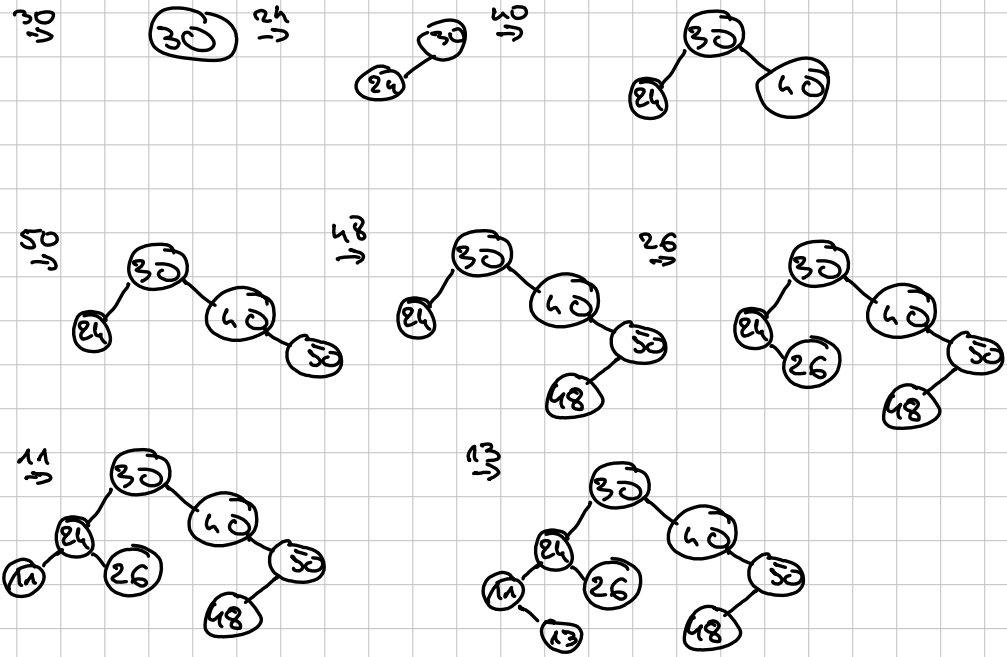


b)
1. $1/1 = 1 > \frac{3}{4}$
2. $2/2 = 1 > \frac{3}{4}$
3. $4/4 = 1 > \frac{3}{4}$
4. $5/8 < \frac{3}{4}$



Vaje 4

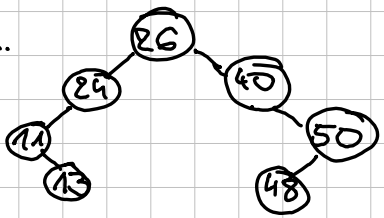
1. Ustavi naslednje elemente v dvojiško iskalno drevo:
   30, 24, 40, 58, 48, 26, 11, 13



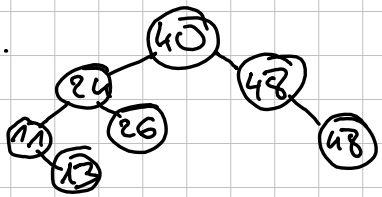
Izbriši 30 iz zadnjega drevesa.

1. Zamenjamo ga lahko z največjim el. v levem poddrevesu
2. Zamenjamo ga lahko z najmanjšim el. v desnem poddrevesu

1.



2.



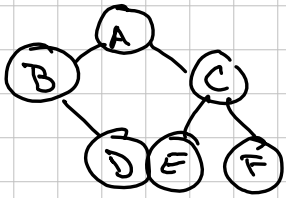
Naredi premi, vmesni in obratni pregled drevesa.

Premi (KLD): 30, 24, 11, 13, 26, 40, 58, 48

Vmesni (LKD): 11, 13, 24, 26, 30, 40, 48, 58

Obratni (LDK): 13, 11, 26, 24, 48, 58, 40, 30

2. Dano je naslednje iskalno drevo:



a) Poišči vmesni pregled drevesa:

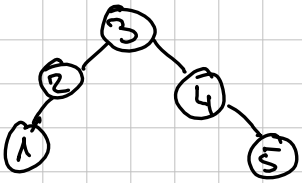
B, D, A, E, C, F

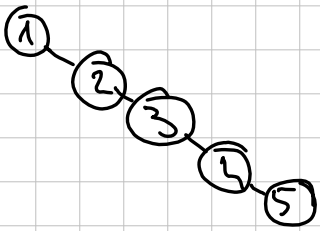
b) Opiši alg., ki zgradi dvojiško drevo iz danega vmesnega in premega pregleda.

1. Za koren vzemi prvi element v premem pregledu
   x = Premi[minPR]
2. Poišči i, da Vmesni[i] = x
3. Rekurzivno zgradi levo poddrevo na
   Premi[minPR+1, ..., i - minVM]
   Vmesni[minVM, ..., i - 1]
   in desno poddrevo na

Premi[i-minVM+1, ..., maxPR]
Vmesni[i+1, ..., maxVM]

c) Pokaži da vmesni pregled ne zadošča, da lahko rekonstruiramo drevo

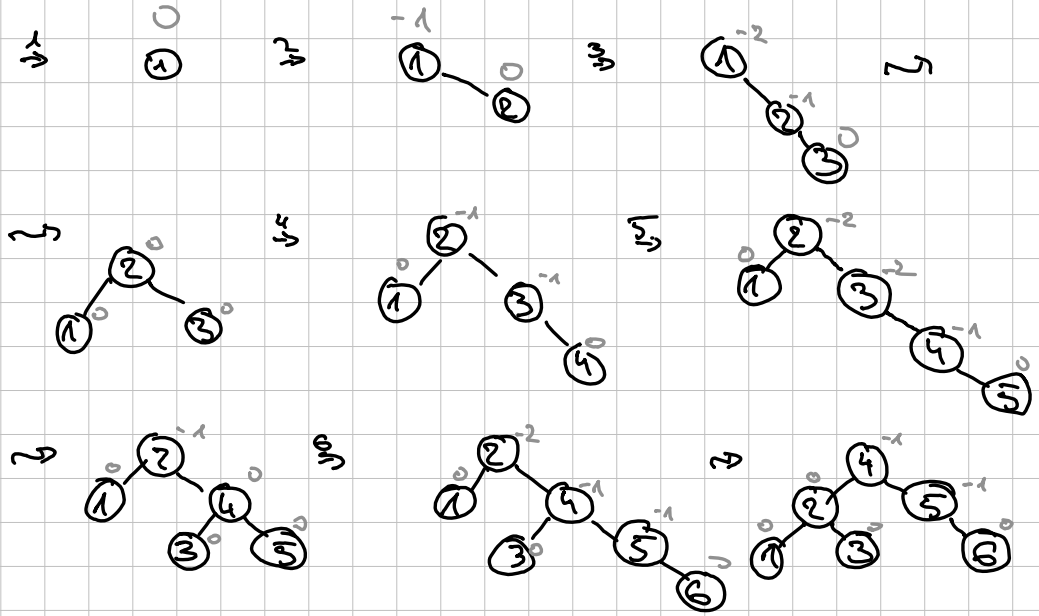




Vmesni: 1,2,3,4,5 1,2,3,4,5

3. Vstavi v AVL drevo zaporedna št. 1-7.

Uravnoteženost (L-D):



Če je $n = 2^k - 1$ za nek k, kako izgleda AVL drevo z elementi od 1 do n.

Vaje 5 — 8.11.24

1. Uporabi dvojiška drevesa za urejanje n št. in opiši ter analiziraj časovno zahtevnost takšnega algoritma.

   Koraki: → AVL je boljši način
   1) Zgradi (dvojiško iskalno drevo) iz teh n elementov
   2) Naredi vmesni pregled na zgrajenem drevesu

   $O(n)$ ↓ $O(n \log n)$

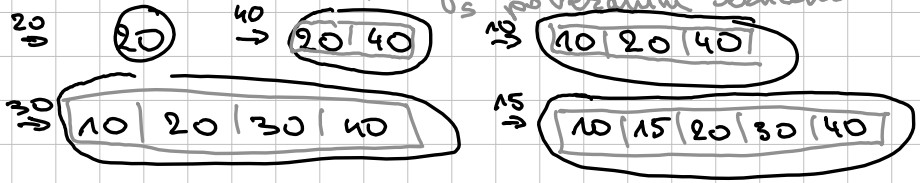
   $\log(1)$
   $\log(2)$
   ⋮
   $\log(n) = O(\log(n))$

   $\Sigma \Rightarrow O(n \log n)$

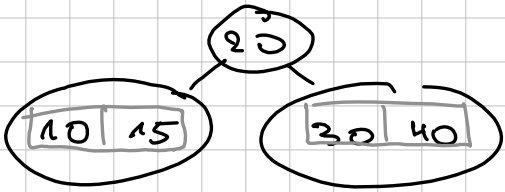
   $1 + 2 + 3 + 4 + 5 + \dots + n \leq \frac{n(n+1)}{2} \Rightarrow O(n^2)$

2. Ustavi naslednje elemente v B drevo reda $b=5$:
   20, 40, 10, 30, 15, 35, 7, 26, 18, 22, 5, 42, 13, 46, 27, 8, 32, 24, 45, 25

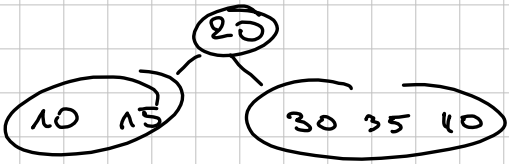
   ⇒ Najlažje taka vozlišča predstavimo s povezanim seznamom



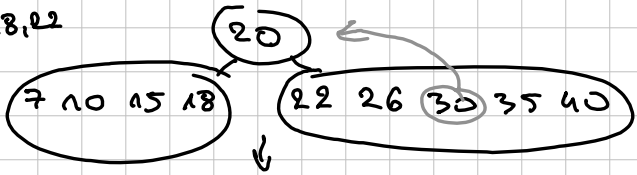
15 →

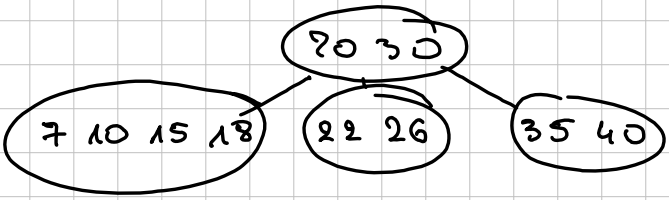


35 →

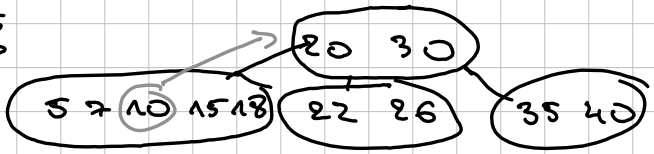


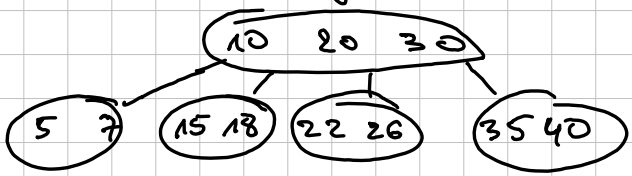
7, 26, 18, 22 →



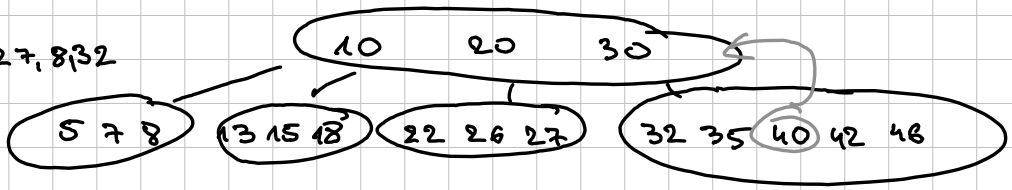


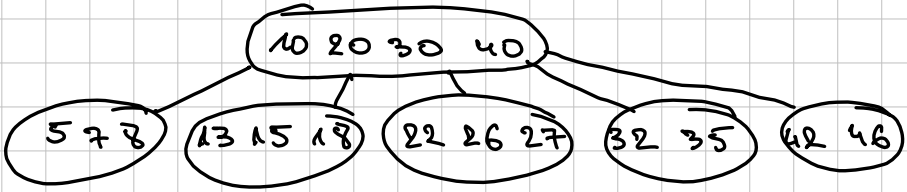
5 →



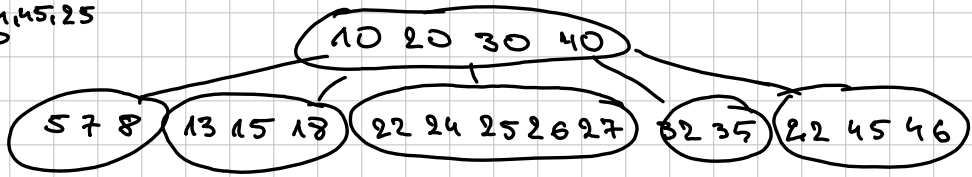


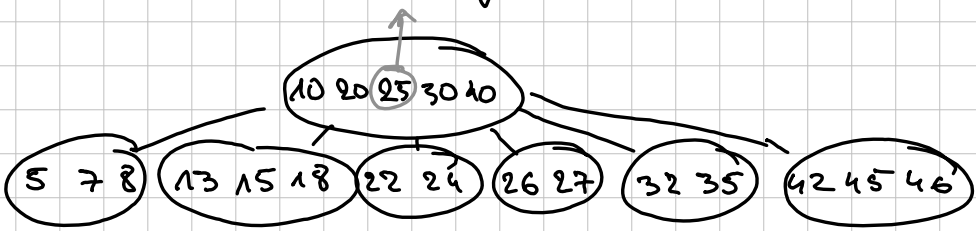
42, 13, 46, 27, 8, 32 →

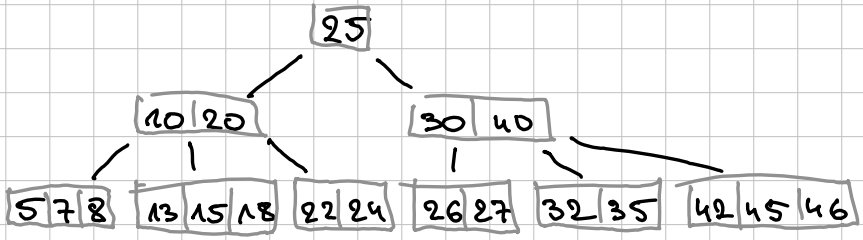




24, 45, 25 →

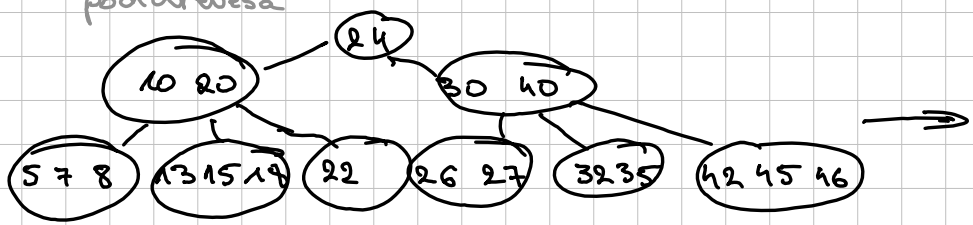




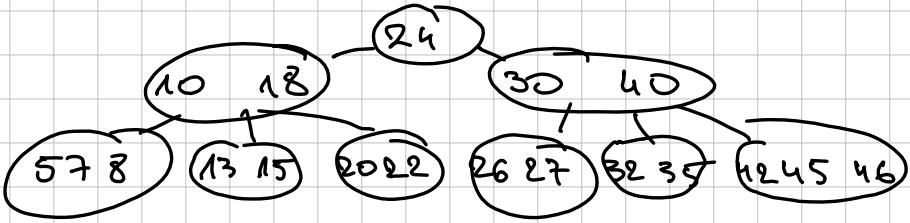


3. Iz zadnjega B-drevesa izbriši 25, 45, 24, 32.

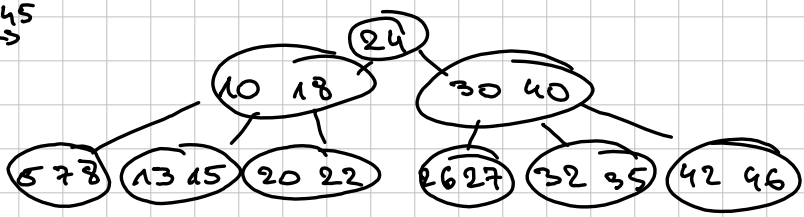
1) -25 za novo vozlišče določimo najmanjši el. iz desnega poddrevesa ali največji el. iz levega poddrevesa



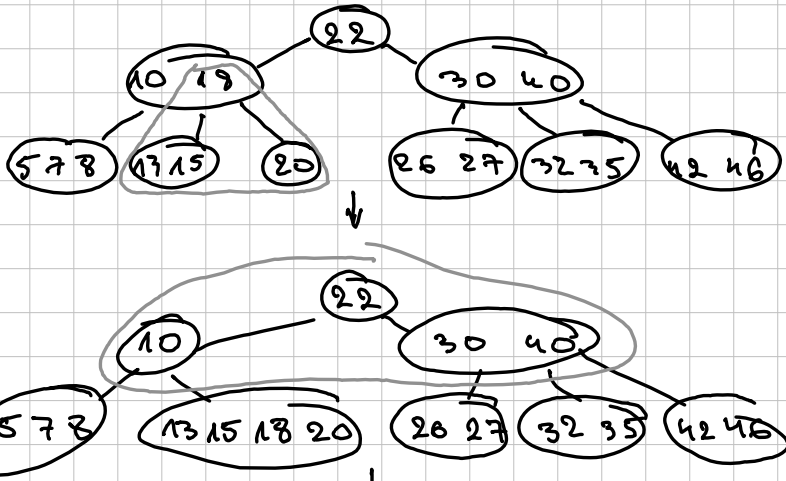
Če je v listu le 1 el. potem si sposodimo, ga nesemo nivo višje in max el. iz korena nesemo v pravi list.


24
10 18
30 40
5 7 8
13 15
20 22
26 27
32 35
42 45 46

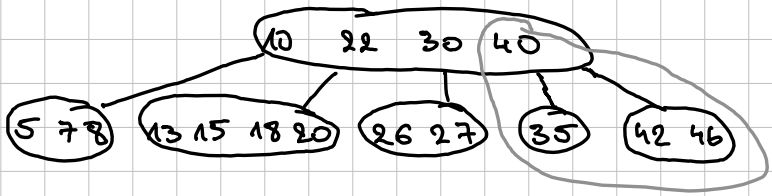
-45

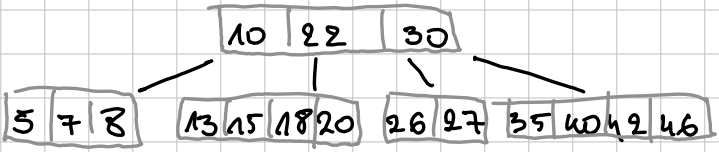

24
10 18
30 40
5 7 8
13 15
20 22
26 27
32 35
42 46

-24

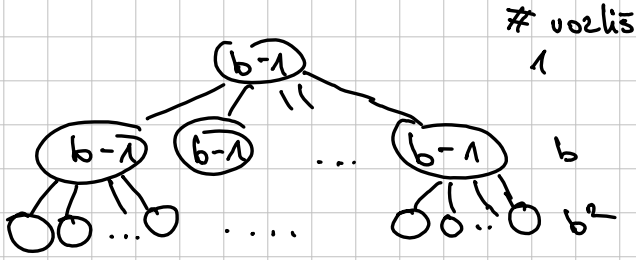

22
10 18
30 40
5 7 8
13 15
20
26 27
32 35
42 46
22
10
30 40
5 7 8
13 15 18 20
26 27
32 35
42 46
10 22 30 40
5 7 8
13 15 18 20
26 27
32 35
42 46

-32


10 22 30 40
5 7 8
13 15 18 20
26 27
35
42 46



4. Največ koliko elementov ima lahko B-drevo reda b z višino h.



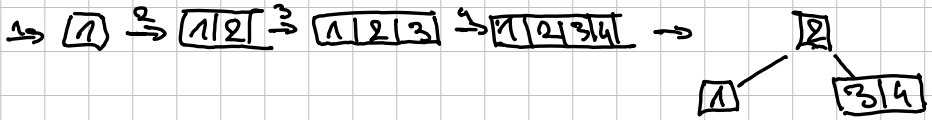
Višina:	# elementov
1	$(b-1)$
2	$b(b-1)$
3	$b^2(b-1)$
⋮	⋮
i	$b^{i-1}(b-1)$
⋮	⋮
h	$b^{h-1}(b-1)$

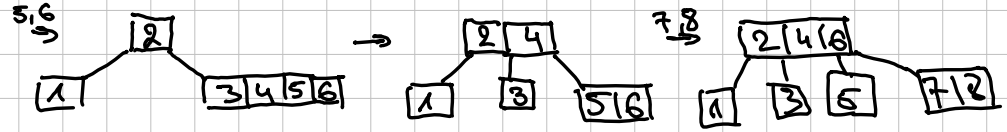
max # el.

$$\sum_{i=1}^{h} b^{i-1}(b-1) = (b-1)\sum_{i=1}^{h} b^{i-1}$$

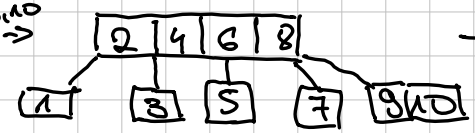
$$= (b-1)\frac{b^h-1}{b-1} = b^h-1$$

5. Vstavi števila od 1 do 15 v drevo reda b=4. Recimo, da vstavimo št. od 1 do n v drevo reda b=4. Opiši, kako izgleda takšno drevo. Kakšna je njegova višina

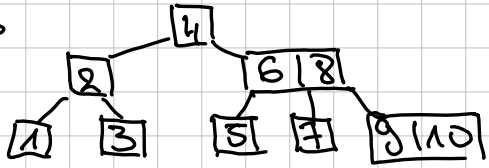




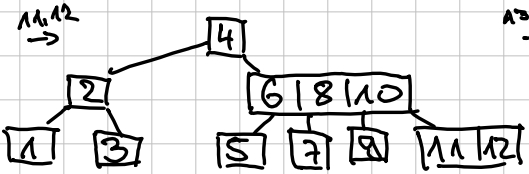
9,10 →



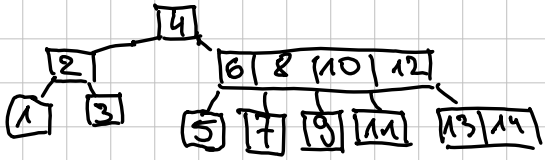
→



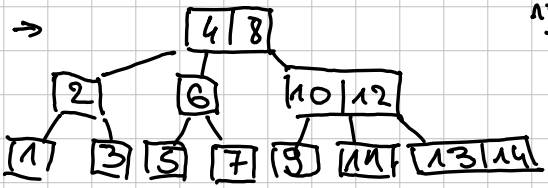
11,12 →



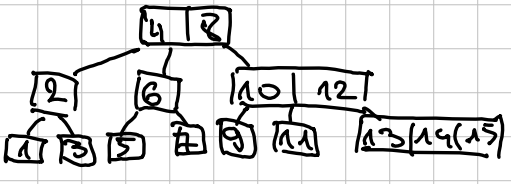
13,14 →



→



15 →



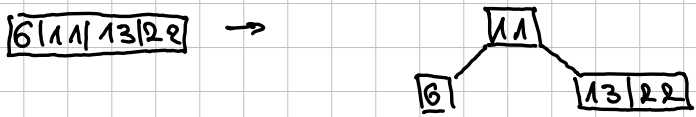
Vsak najmanjši element v korenu lahko izračunamo po formuli $2^{(\log n)}$

Vaje 6

15.11.24

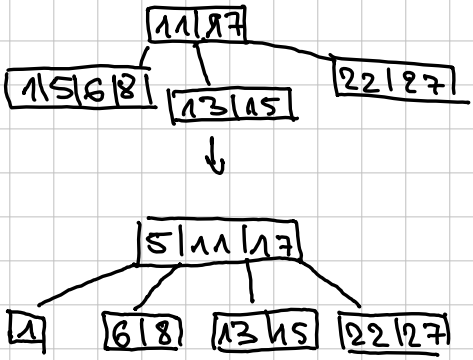
1. Vstavi naslednje elemente v B-drevo reda b=4:
   22, 11, 6, 13, 17, 27, 8, 15, 5, 1

27,11,6,13 →

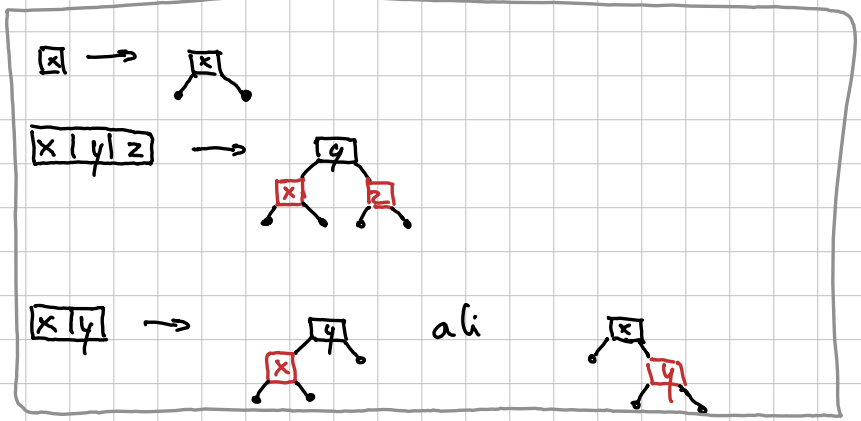


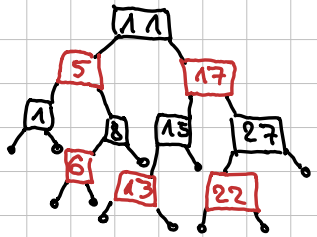
17,27 →

8, 15, 5, 1



2. Pretvori zadnje B-drevo u rdeče-črno drevo

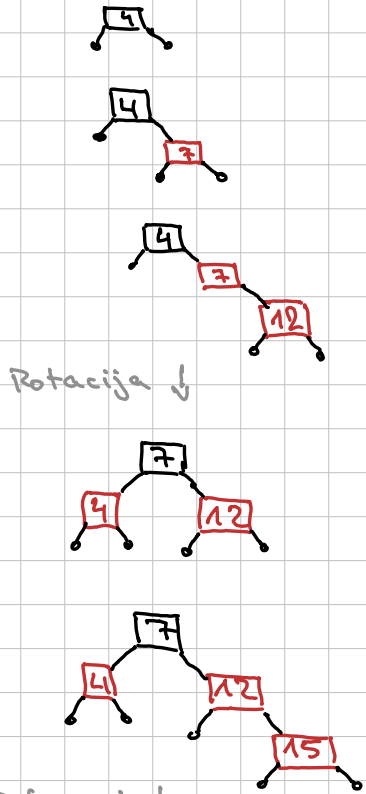




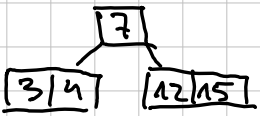
3. Vzporedno vstavi naslednje elemente v rdeče-črno drevo in pripadajoče B-drevo, b=4:
4, 7, 12, 15, 3, 5, 14, 18, 16, 17

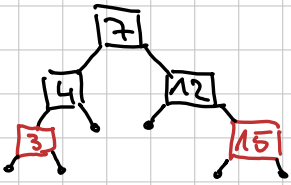
B-drevo

R-Č drevo

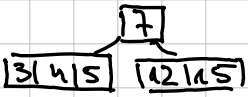


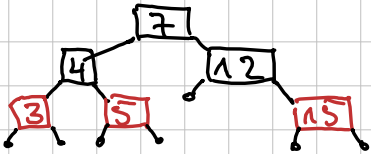
3 →



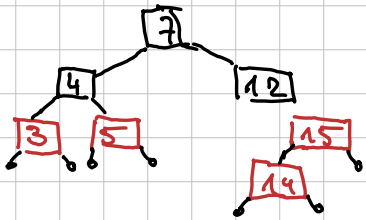


5 →

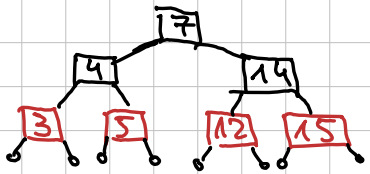




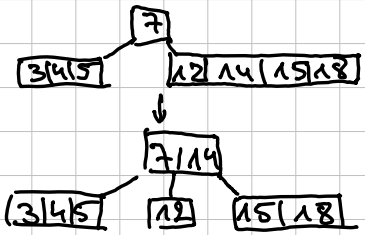
14 →

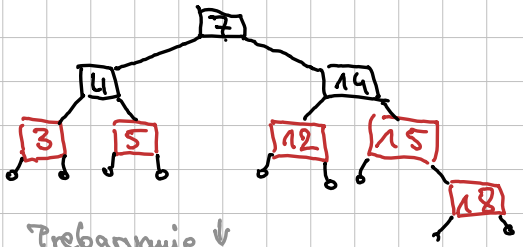


Rotacija ↓

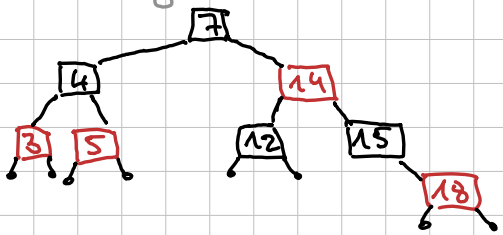


18 →

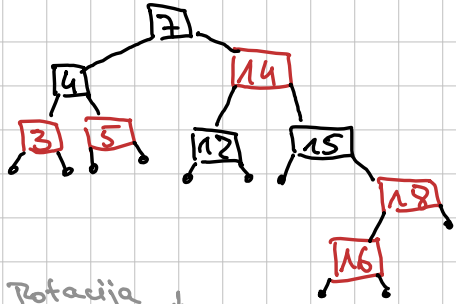




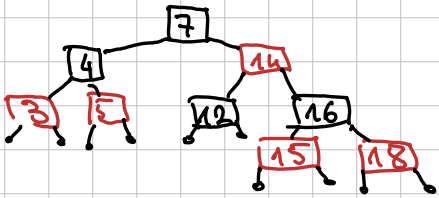
Prebarvanje ↓



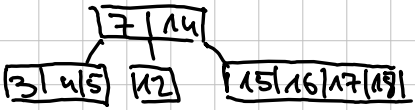
16

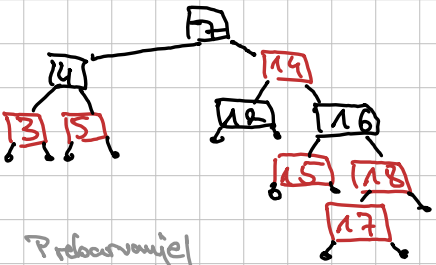


Rotacija

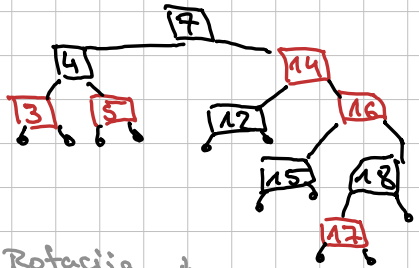


17

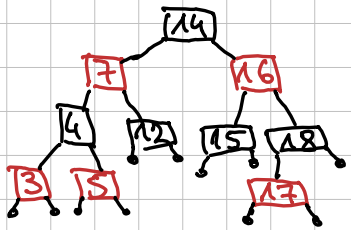




Preobrnjeno



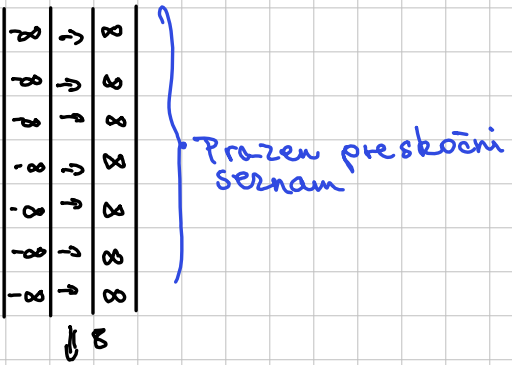
Rotacija

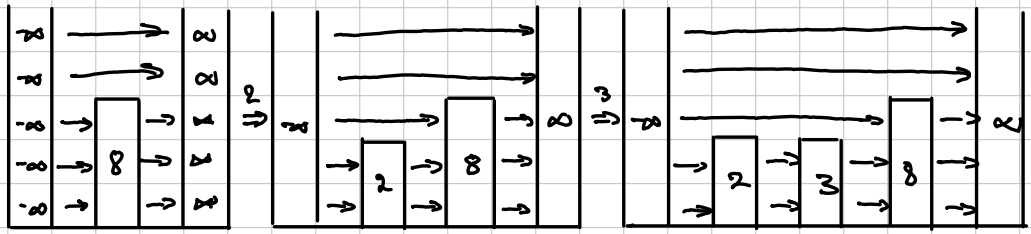


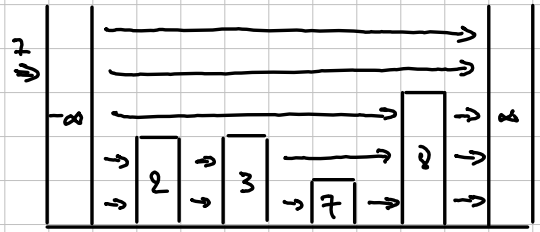
4. V preskočni seznam vstavi naslednje elemente:
8, 2, 3, 7, 13, 19, 12, 26; kjer generator naključnih št. vrne naslednje zaporedje:

1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, ...

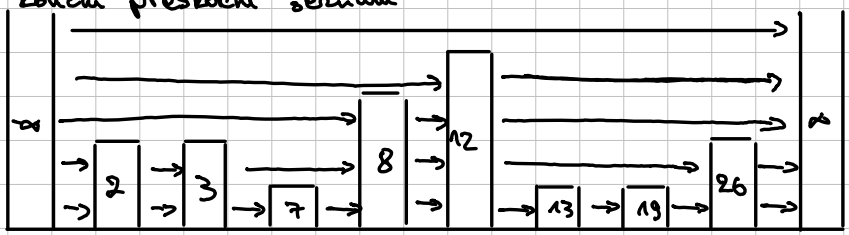
8 2 3 7 13 19 12 26







končni preskočni seznam



Izbriši 13 in 2 iz končnega pres. seznama.