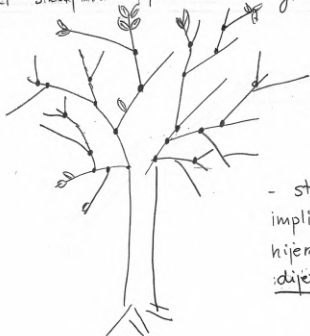


STABLA

- dinamički skupovi koji ulaze u grupu nelinearnih struktura podataka

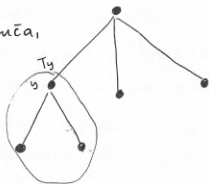


- stablo je povezan aciklički graf

- struktura podataka koja implicitno podrazumijeva hijerarhijski odnos roditelj - dijete

- svojstva stabla: T_0

- svaki čvor x sadrži ključ $key - x.key$
- postoji specijalan element zvan korijen stabla - $T.root$
- svaki nekorijenski čvor predstavlja novi korijenski čvor za podstablo T
- svaki čvor x , osim ključa, sadrži pokazivače na roditelja i na djecu



- Korijen je element koji nema roditelja
- list je element koji nema djece

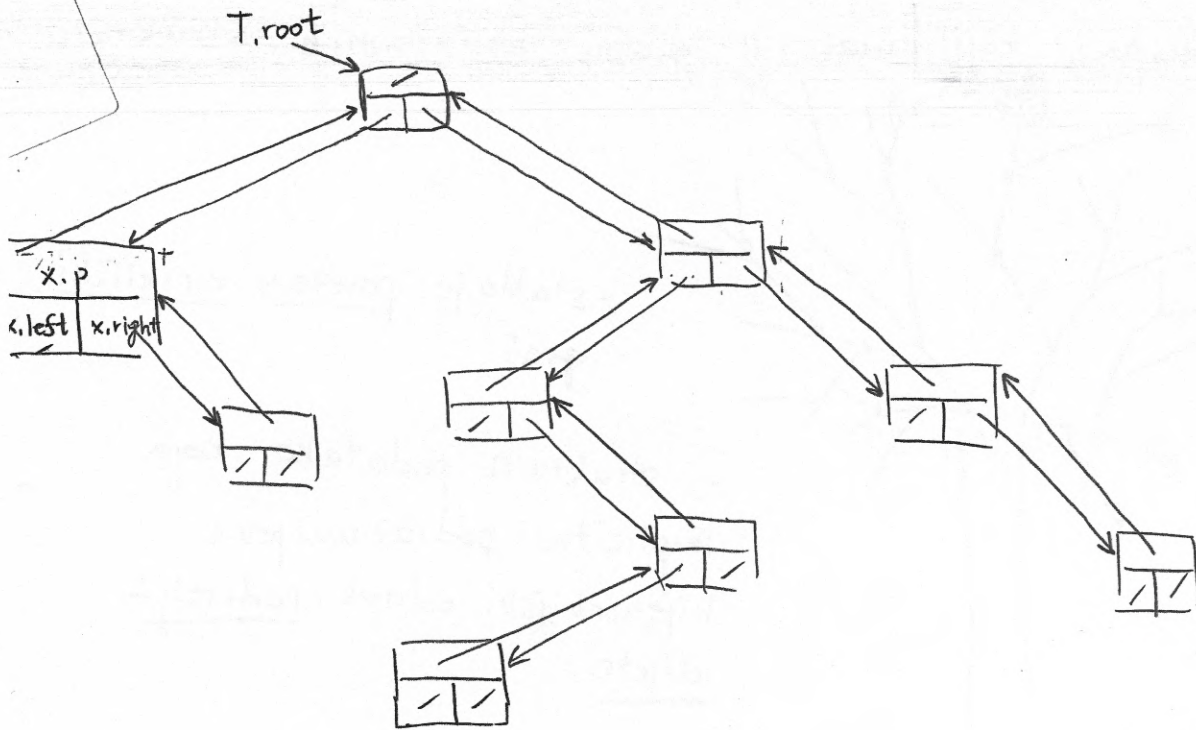
stupanj čvora = broj podstabala pri čvoru u danom stablu

stupanj stabla = maksimalan stupanj čvora u stablu

stablo je strukturirano po nivoima

dubina/visina stabla je maksimalan broj nivoa u stablu

BINARNA STABLA



1. Stabila je 2

no stablo T je povezana struktura podataka koja
elemente i pokazuje na korijen $T.root$
element x stabla T sadrži:

- `x.p` - pokazivač na roditelja
- `x.left` - pokazivač na lijevo dijete od `x`
- `x.right` - pokazivač na desno dijete od `x`

$x.p = NIL$ onda je x korijen stabla

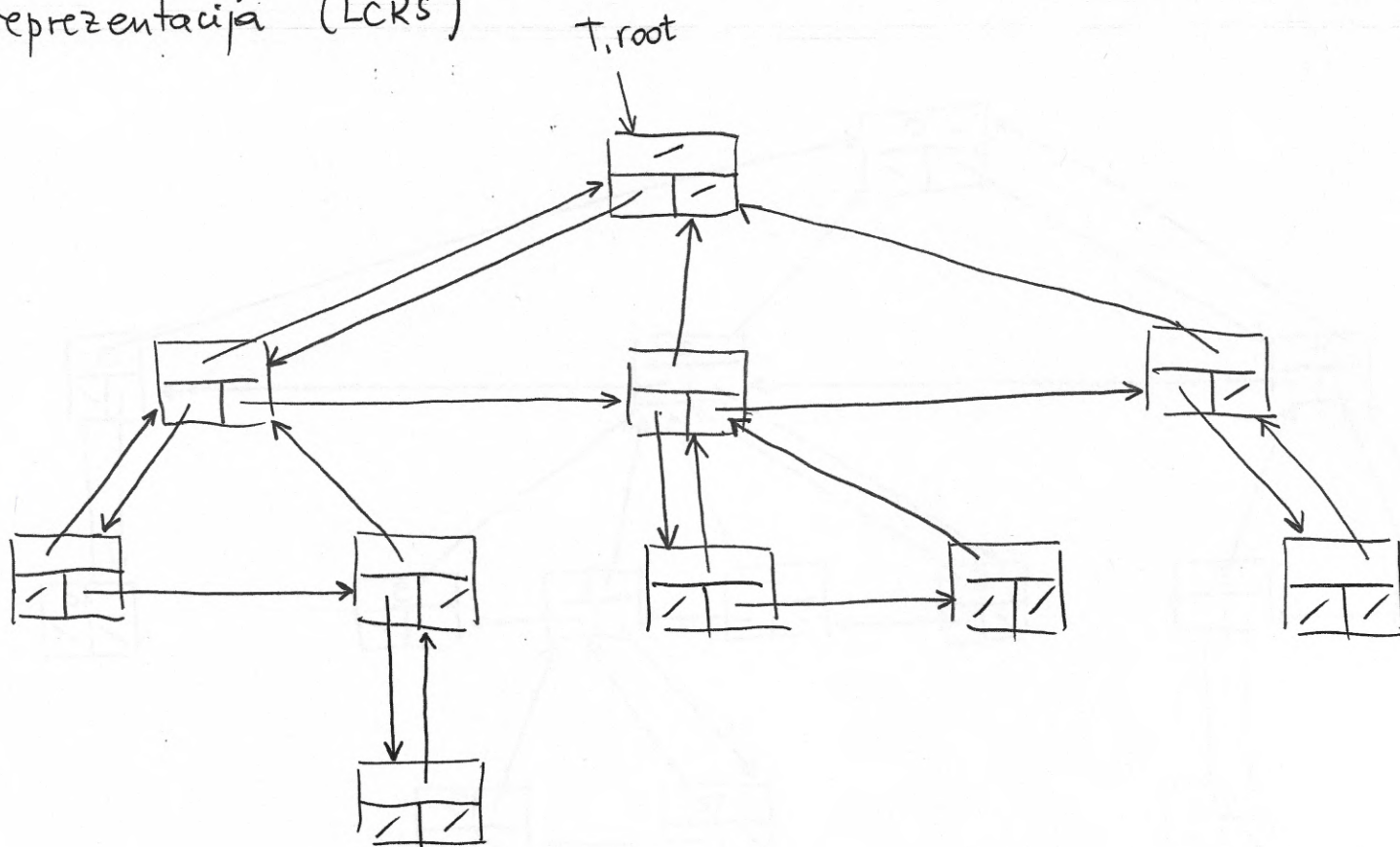
nema lijevo (desno) djeteta onda je $x.\text{left} = \text{NIL}$

$$t = \text{NIL})$$

$T.root = NIL$ onda je stablo T prazno

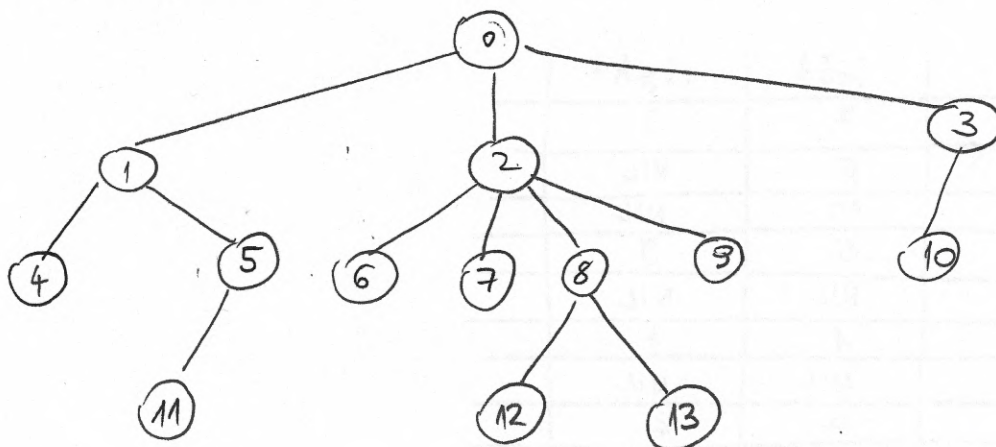
STABLA NEOGRANIČENOG STUPNJA

- LEFT-CHILD-RIGHT-SIBLING (Lijero - dijete - desni-brat)
reprezentacija (LCRS)

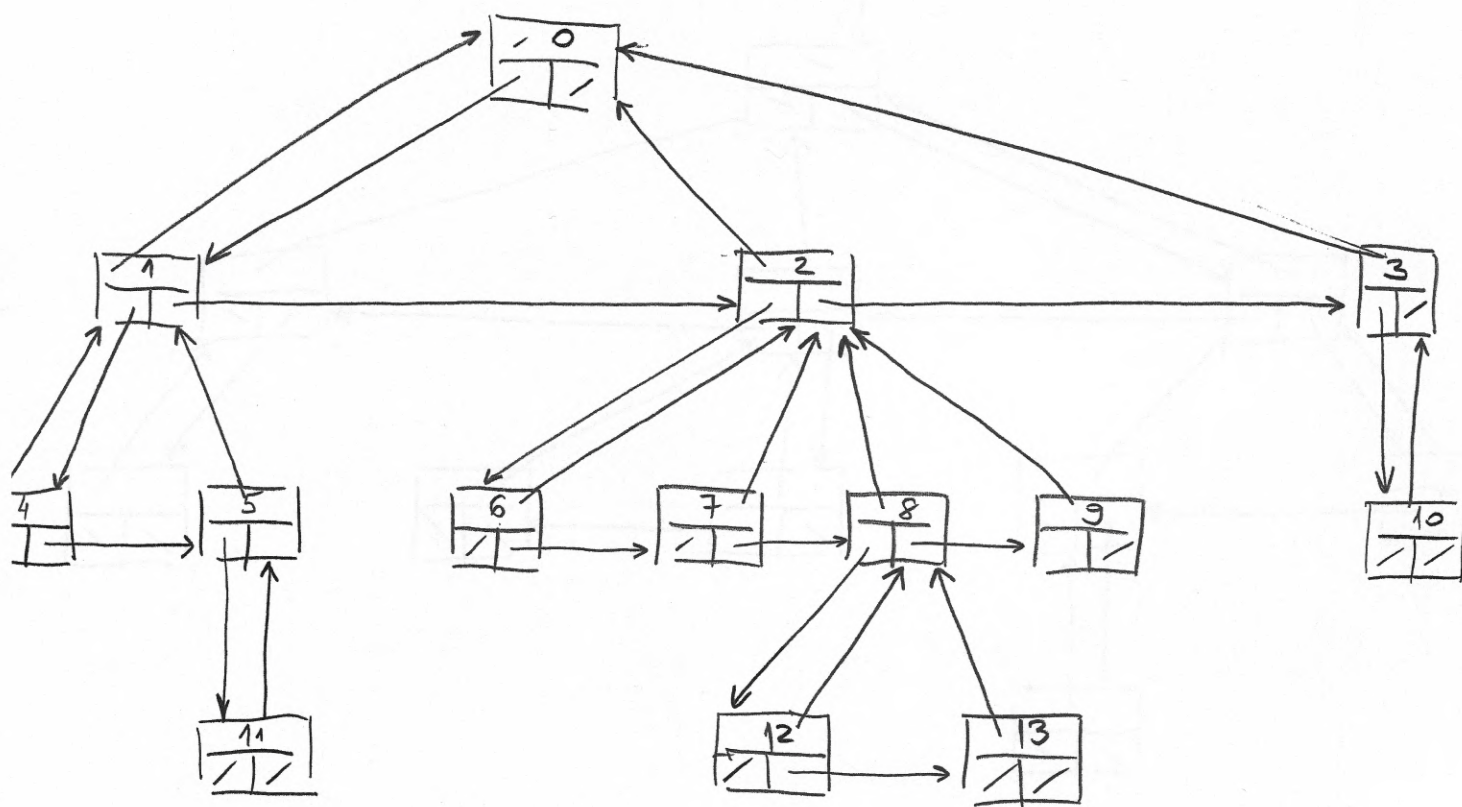


- stupanj stabla nije ograničen
- stablo T sadrži pokazivač na korijen T.root
 - x.p - pokazivač na roditelja
 - x.left-child - pokazivač na lijevo dijete od x
 - x.right-sibling - pokazivač na desnog brata od x

Primjer: Dano je stablo T na sljedećoj slici:



1. tvrdite $x.p$, $x.left-child$, $x.right-sibling$, dubinu/visinu stabla
 2. listove koji je $x.key = 6$ i prikažite ga u LCRS reprezentaciji.



key = 6

left-child = NIL

right-sibling = 7

$d = 2$

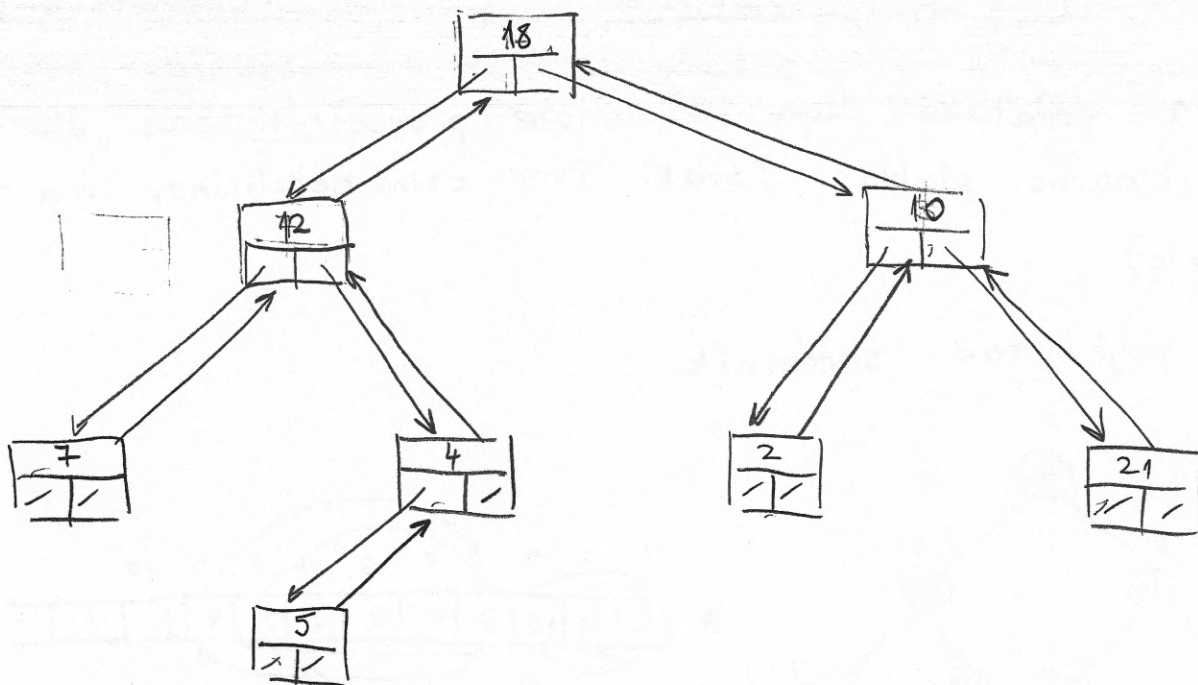
listovi: 4, 11, 6, 7, 12, 13, 9, 10

dubina/visina: 3

najer:

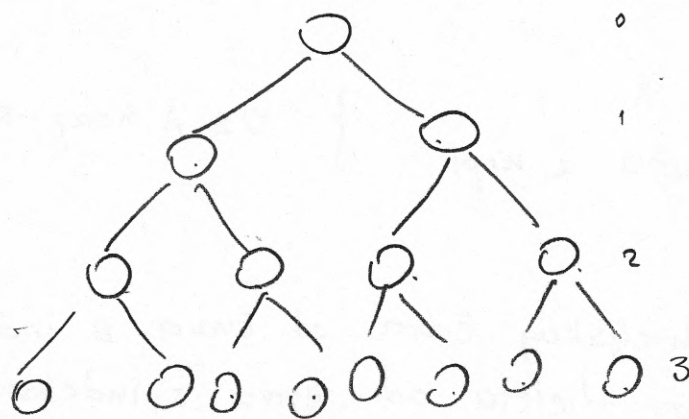
Nacrtajte binarno stablo s korijenom indeksa 6.
 datu jenom sljedećom tablicom

index	key	left	right
1	12	7	3
2	15	8	NIL
3	4	10	NIL
4	10	5	9
5	2	NIL	NIL
6	18	1	4
7	7	NIL	NIL
8	14	6	2
9	21	NIL	NIL
10	5	NIL	NIL



STABLA BINARNOG PRETRAŽIVANJA

- Vrijeme izvršavanja svih osnovnih operacija proporcionalna je visini stabla (u slučaju potpunih binarnih stabala $\Theta(\lg n)$)



$$\Theta(\lfloor \lg n \rfloor)$$

BINARNA STABLA PRETRAŽIVANJA

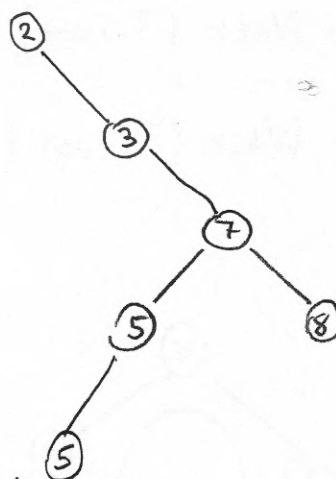
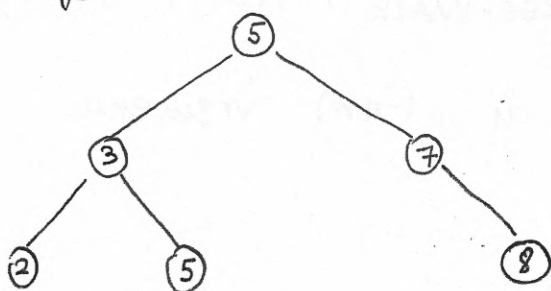
- binarna stabla koja imaju sljedeće svojstvo.

Neka je x čvor u binarnom stablu T .

Ako je y čvor sadržan u lijevom podstablu od x , onda je $y.key \leq x.key$. Ako je

y čvor sadržan u desnom podstablu od x , onda je $x.key \leq y.key$.

Primjer:



- obilasci binarnog stabla pretraživanja.

- * INORDER - lijevo podstablo + korijen + desno podstablo
- * PREORDER - korijen + lijevo podstablo + desno podstablo
- * POSTORDER - lijevo podstablo + desno podstablo + korijen

INORDER-TREE-WALK (x)

if $x \neq NIL$

then INORDER-TREE-WALK ($x.left$)

print $x.key$

INORDER-TREE-WALK ($x.right$)

PREORDER-TREE-WALK (x)

if $x \neq NIL$

then

print $x.key$

PREORDER-TREE-WALK ($x.left$)

PREORDER-TREE-WALK ($x.right$)

POST ORDER-TREE-WALK (x)

if $x \neq \text{NIL}$

then POSTORDER-TREE-WALK (x.left)

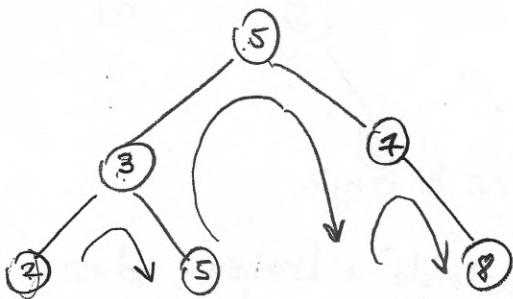
POSTORDER-TREE-WALK (x.right)

print x.key

Theorem. (Same INORDER-TREE-WALK)

Neka je T stablo od n čvorova. Tada se obilasci
INORDER-TREE-WALK ($T.\text{root}$), PREORDER-TREE-WALK ($T.\text{root}$);
POSTORDER-TREE-WALK ($T.\text{root}$) izvršavaju u $\Theta(n)$ vremenu.

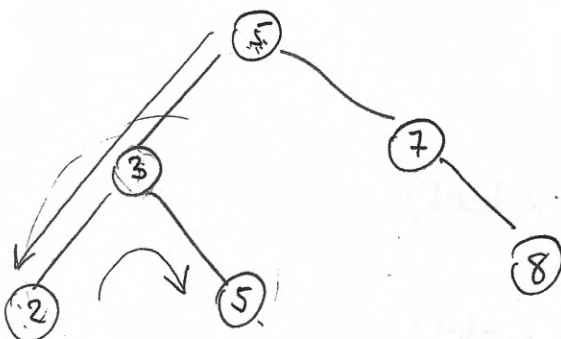
Primjer:



INORDER:

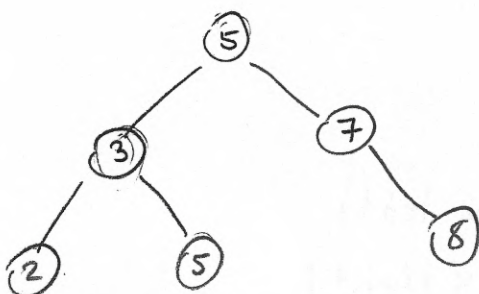
2 3 5 5 7 8

Sortirani brojevi
po ključu



PREORDER:

5 3 2 5 7 8

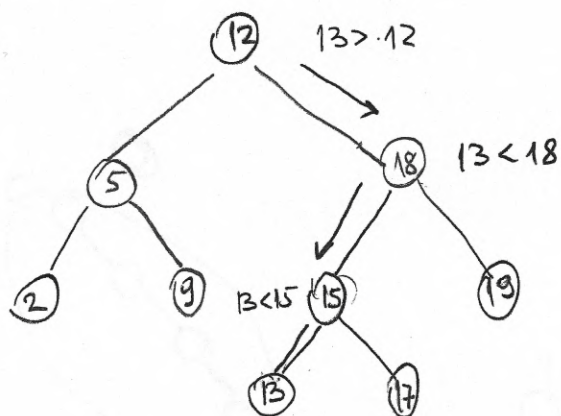


POSTORDER:

2 5 3 8 7 5

UBACIVANJE ELEMENATA U BINARNO STABLO PRETRAŽIVANJA

Primjer:



Ubaciti čvor s ključem 13

1. pronaći odgovarajuću poziciju
2. ubaciti element

TREE-INSERT (T, z)

$y \leftarrow \text{NIL}$

$x \leftarrow T.\text{root}$

while $x \neq \text{NIL}$

$y \leftarrow x$

if $z.\text{key} < x.\text{key}$
 then $x \leftarrow x.\text{left}$
 else $x \leftarrow x.\text{right}$

end while

$y \leftarrow z.p$

if $y = \text{NIL}$ (ako je T bilo prazno stablo)

then $T.\text{root} = z$

else if $z.\text{key} < y.\text{key}$
 then $y.\text{left} \leftarrow z$
 else $y.\text{right} \leftarrow z$

end if

TREE-PREDECESSOR (x)

```
if x.left  $\neq$  NIL  
|  
return TREE-MAXIMUM (x.left)  
end if
```

} $O(n)$

$y \leftarrow x.p$

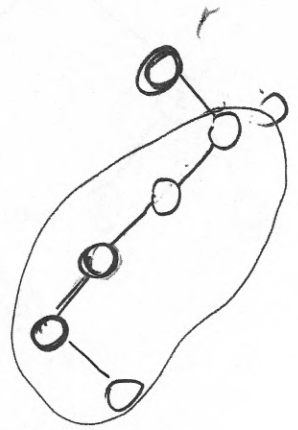
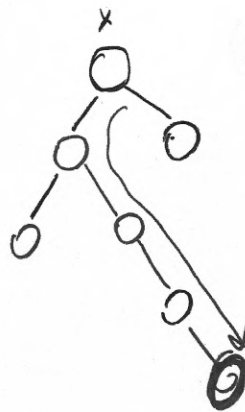
while $y \neq \text{NIL}$ and $x = y.\text{left}$

$x \leftarrow y$

$y \leftarrow y.p$

end while

return y



Minimalni i maksimalni element

Ulaž: korijen podstabla x

Izlaz: pokazivač na čvor s (najmanjim / najvećim) ključem u stablu s korijenom x

TREE-MINIMUM (x)

while $x.\text{left} \neq \text{NIL}$

$x \leftarrow x.\text{left}$

end while

return x

TREE-MAXIMUM (x)

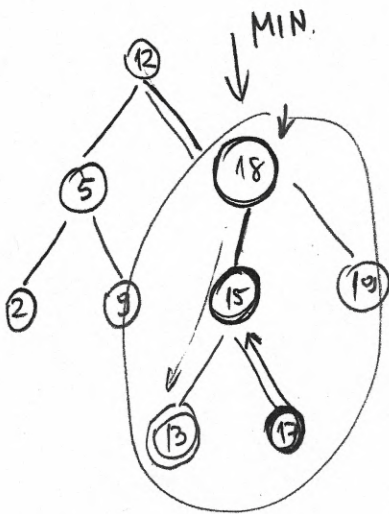
while $x.\text{right} \neq \text{NIL}$

$x \leftarrow x.\text{right}$

end while

return x

Sljedbenik čvora



sljed. 12 je 13

sljed od 17 je 18

moguća su dva slučaja:

- 1.) čvor x nema desno podstablo
- 2.) ima — — — — — ✓

Propozicija

Neka je T BSP s čvorovima čiji su ključevi međusobno različiti.

Ako je desno podstablo s korijenom x u T prazno i x ima sljedbenika y , onda je y prvi potomak (predak) od x čiji je lijevo dijete također predak od x .

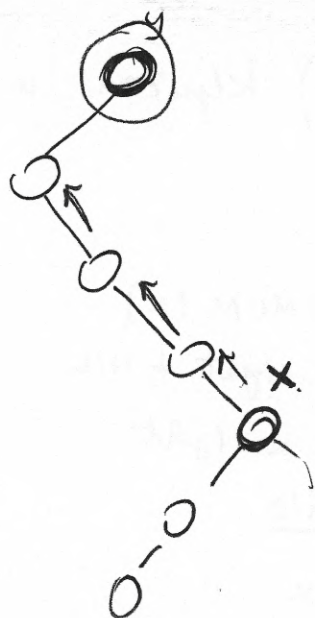
kaz: Ako je y sljedbenik od x onda je x prethodnik od y .

Lijevo podstablo od y je neprazno.

x je maksimalni element u lijevom podstablu od y .
Ako pogledamo TREE-MAXIMUM (y) uvidjet ćemo

ie. u, istinitost tvrdnje.

□



Tree-Successor (x)

```

if x.right ≠ NIL
    return Tree-Minimum (x.right)
end if
y ← x.p
while y ≠ NIL and x = y.right
    x ← y
    y ← y.p
end while
return y
    
```

} $O(n)$

} $O(n)$

Impliza vremena izvršavanja:

Tree-Search: $O(n)$

Tree-Minimum: $O(n)$

Tree-Maximum: $O(n)$

Tree-Successor: $O(n)$

Tree-Predecessor: $O(n)$

Zaključak: Neka je T BSP visine n . Tada se operacije SEARCH, MINIMUM, MAXIMUM, SUCCESSOR i PREDECESSOR izvršavaju u vremenu $O(n)$.

Teorija

Ako je x korijen binarnog stabla koje ima n čvorova,
INORDER-TREE-WALK(x) izvršava se u $\Theta(n)$ vremenu.

Dokaz: (metodom supstitucije)

$T(n)$ ← vrijeme izvršavanja

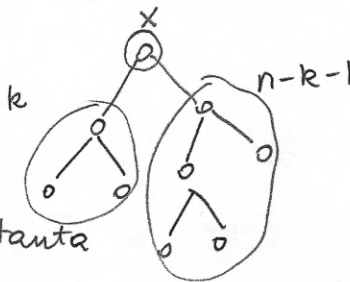
$n=0 \Rightarrow T(0) = c$, c konstanta

$n>0 \Rightarrow$

$x \neq \text{NIL}$

$$T(n) = T(k) + T(n-k-1) + d$$

d konstanta



Metoda supstitucije (met. indukcija)

$$T(n) = \Theta(n) = (c+d)n + c$$

$n=0$

$$T(0) = (c+d) \cdot 0 + c = c \quad \checkmark$$

Pretpostavka: Tvrdnja vrijedi za n , tj:

$$T(n) = (c+d) \cdot n + c \quad //$$

korak:

$$0 < k < n$$

$$\begin{aligned} T(n+1) &= T(k) + T(n+1-k-1) + d \\ &= T(k) + T(n-k) + d \\ &= (c+d)k + \underline{c} + (c+d)(n-k) + c + \underline{d} \\ &= (c+d)(k+1) + (c+d)(n-k) + c \\ &= (c+d)(n-k+k+1) + c \\ &= (c+d)(n+1) + c \end{aligned}$$

$$\Rightarrow T(n) = \Theta(n)$$

□

pregled, operacija:

- obilasci BSP-a: - INORDER
 - POSTORDER
 - PREORDER
- pretraživanje
 - minimum i maksimum
 - sledbenik i prethodnik
 - umetanje i brisanje

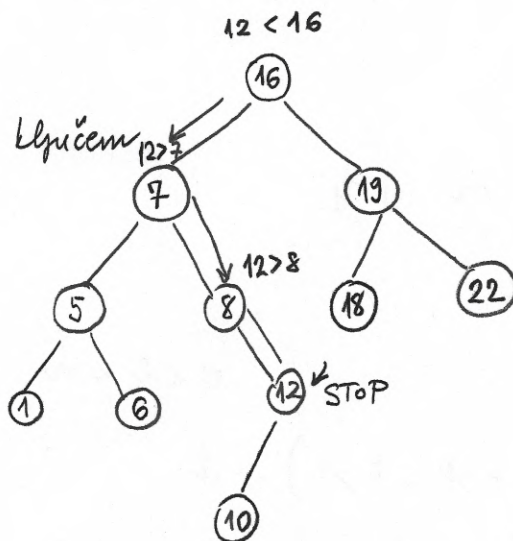
pretraživanje:

laz: korijen x ključ k

laz: pokazivač na čvor u T kojemu je ključ k, ili NIL ukoliko takav ne postoji

primer:

uzimo čvor s ključem 12



data:

Napiši iterativnu verziju.

ITERATIVE-TREE-SEARCH(x, k)

while $x \neq \text{NIL}$ and $k \neq x.\text{key}$

if $k < x.\text{key}$

$x \leftarrow x.\text{left}$

else

$x \leftarrow x.\text{right}$

end if

end while

return x

Uputa: koristi sugstro
BSP

TREE-SEARCH(x, k)

if $x = \text{NIL}$ or $k = x.\text{key}$

return x

end if

if $k < x.\text{key}$

TREE-SEARCH(x.left, k)

else

TREE-SEARCH(x.right, k)

end if