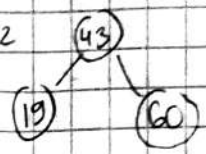
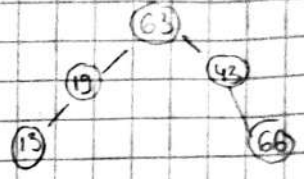
Dz 3

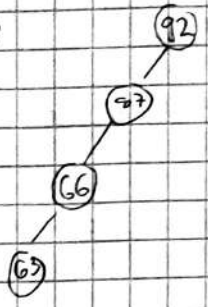
1) h=2



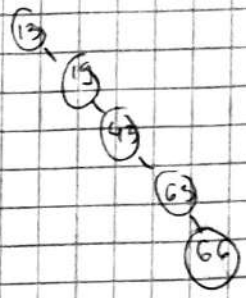
h=3



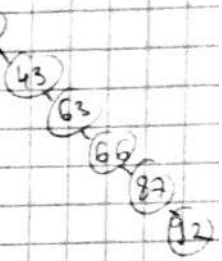
h=4



h=5



h=6



2) Dokaz

Po svojstvu BST-a znamo da y mora biti predak od x.
Da nije imali bi neki z najbliži zajednički predak od x i y
pa bi imali $x < z < y \Rightarrow$ y nije sljedbenik.

Dali je y najbliži? Lijevo dijete od y mora biti predak od x
(da je desno onda bi $x > y$)

Pretpostavimo suprotno, tj. da y nije najniži predak od x
čije je dijete isto predak od x.

Neki čvor z bi bio najniži predak od x čije je lijevo dijete isto
predak od x

No tada z mora biti u lijevom podstablu od y pa je onda
$z < y$ što je kontradikcija s pretpostavkom da je y slj. od x

Također x mora biti čvor s max ključem u lijevom podstablu
od y

$\Rightarrow$ y sljedbenik od x i njegovo dijete predak od x ujedno
i najniži predak od x

3) x je list $\Rightarrow$ nema djece

x može biti lijevo ili desno dijete od y

1° x lijevo dijete y:

x je list $\Rightarrow$ x je prethodnik od y $\Rightarrow$ y slj od x $\Rightarrow$

y. ključ je najmanji ključ u T veći od x. ključ

2° x desno dijete

x list $\Rightarrow$ x sljedbenik od y $\Rightarrow$ y prethodnik od x $\Rightarrow$

y. ključ je najveći ključ u T manji od x. ključ

1. DODAVANJE

```
TREE-INSERT (T, z)   ▷ ubacujemo čvor z
1.  y = NIL
2.  x = T.root
3.  while x ≠ NIL
4.      y = x
5.      if z.key < x.key
6.          x = x.left
7.      else x = x.right
8.  z.p = y
9.  if y = = NIL
10.     T.root = z
11. elseif z.key < y.key
12.         y.left = z
13  else y.right = z
```

VRIJEME: $O(h)$

TRAŽIVANJE

```
TREE-SEARCH (x, k)
1.  if x == NIL or k == key[x]
2.      return x
3   if  k < x.key
4.      return TREE-SEARCH (x.left, k)
5.  else return TREE-SEARCH (x.right, k)
```

4) Neka je x čvor kojeg prosljeđujemo TREE-SUCCESSOR i y k-ti sljedbenik od x (na y završavamo). Udaljenost ta dva čvora je max $2h$ (dva odlaska visine stabla) ove veze koje spajaju ih k čvorova će biti prođene dvaput i trebat $2k + 2h$ tj. $O(k+h)$

5) Vrijeme obilaska TREE-MINIMUM je $O(h)$

U prethodnom zadatku vidimo TREE-SUCCESSOR ima $O(k+h)$ za k poziva

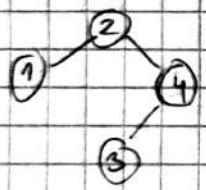
Spec. za $k = n-1$ imamo $O(n-1+h)$, a znamo $n \geq h$ (br čvor $\geq$ visina)
=> vrijeme $O(n)$

6) Oba algoritma se izvršavaju u $O(h)$ vremena

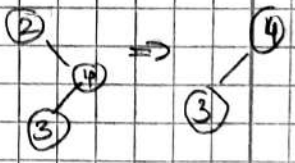
Oba algoritma će proći isti put da dođu do potrebnog čvora

Ali search će proći istim čvorovima i još vratiti nadređeni čvor tj. $\uparrow 1$

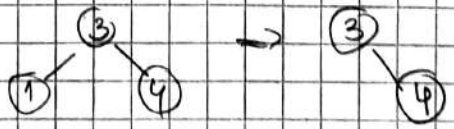
8)



prvo obrnemo 1 pa 2:



obrnuto:



nisu dobivena ista stabla, nije komutativno