AVL Trees – 3/28/22

① Friday class poll
 – Video

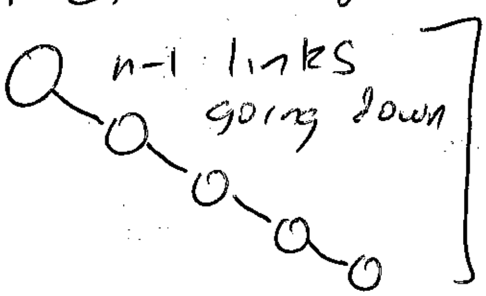
BST runtimes

① go down one path tree – $O(h)$

② go down both L, R → $O(n)$

n = # nodes in tree, h = height

h can range from $\lg n$ to $n-1$.



2^{n-1} different structural trees of n nodes of height $n-1$.

In worst case, BST is no better than a LL.

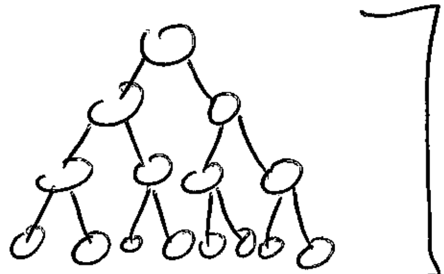
Avg case $h = O(\lg n)$, complicated math to prove, but this is the result.

Best case

$n+1 = 2^{h+1}$

$\log_2(n+1) = h+1$

$h = \log_2(n+1) - 1$



$n = 2^{h+1} - 1$

$n = 1+2+4+\cdots 2^h$
$= 2^{h+1} - 1$

Quite a few balanced binary search trees

Balanced $\Longrightarrow$ $h = O(\lg n)$

In CS1, we teach AVL trees.

Defn: AVL Tree

1. Valid BST.
2. For each node, height left subtree and height right subtree can't differ by more that 1.

$$|h_{left} - h_{right}| \le 1.$$

$\longrightarrow$ AVL tree node property

First, we'll show if we can maintain (1) and (2), $h = O(\lg n)$.

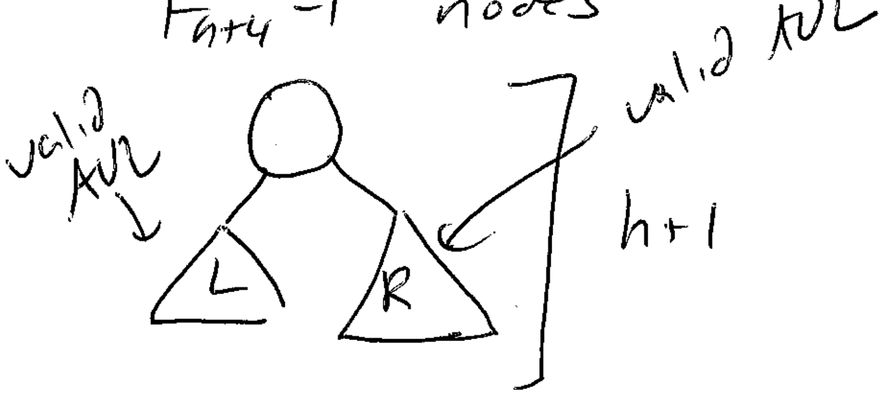
More specifically, we'll prove via strong induction that an AVL tree of height h has at least $F_{h+3} - 1$ nodes, where F_n is the n^{th} Fibonacci #.

b.c. $h=0$ $n=1$ ✓ $F_{0+3} - 1 = F_3 - 1 = 2 - 1 = 1$

$h=1$ $n=2$ ✓ $F_{1+3} - 1 = F_4 - 1 = 3 - 1 = 2$

i.h. Assume ~~for an~~ arbitrarily chosen non-neg int ~~$h = k$~~ that an AVL tree of height k has ~~at least~~ $F_{k+3} - 1$ nodes i.h. Assume for all non-neg int $h \le k$ where $k \ge 1$ that an AVL tree of height h has at least $F_{h+3} - 1$ nodes.

i.s. Prove for $h+1$ that an AVL tree of height $h+1$ must have at least $F_{h+4} - 1$ nodes



$$\max(h_L, h_R) = h$$
$$\min(h_L, h_R) \ge h-1$$

$$
\begin{aligned}
\#\text{ nodes} &= 1 + \#\text{node}(L) + \#\text{node}(R) \\
&\ge 1 + \#\text{nodes}(\text{AVL tree height } h) \\
&\qquad + \#\text{ nodes}(\text{AVL tree height } h-1) \\
&\ge 1 + (F_{h+3} - 1) + (F_{h+2} - 1) \\
&= F_{h+4} - 1
\end{aligned}
$$

$$F_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

ϕ (golden ratio)

$$n \geq F_{h+3} - 1$$

$$n \geq \frac{1}{\sqrt{5}} \phi^n - 1$$

$$n+1 \geq \frac{1}{\sqrt{5}} \phi^n$$

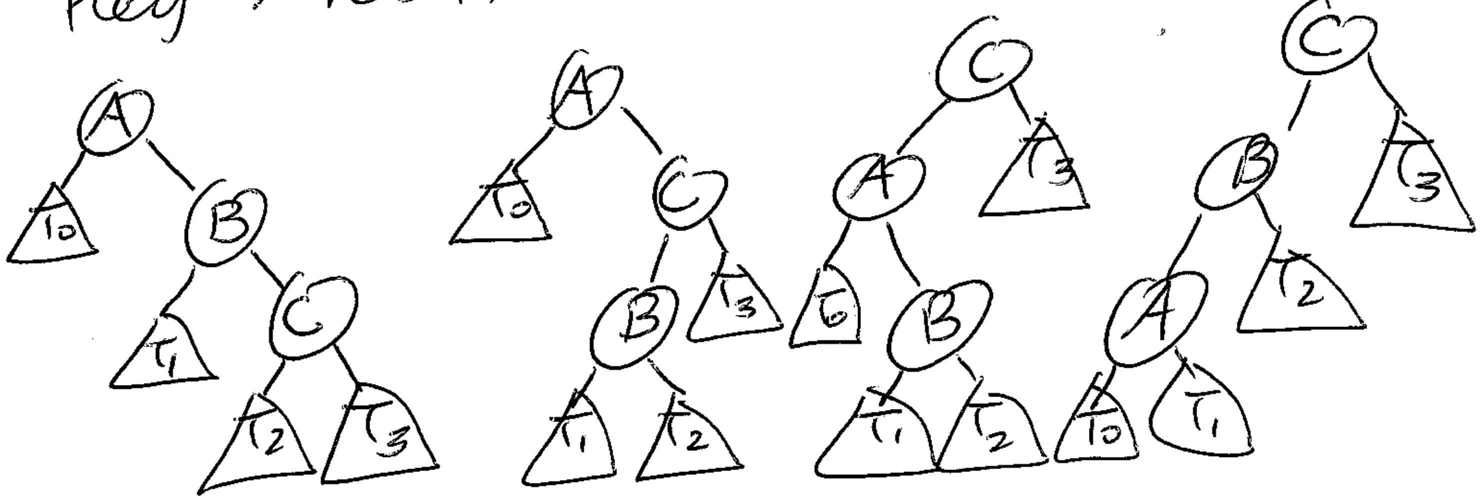
$$\sqrt{5}(n+1) \geq \cancel{\phi(n)}\, \phi^n$$

$$\log_\phi \sqrt{5}(n+1) \geq n \implies n \leq \overbrace{\log_\phi \sqrt{5}}^{\text{const}} + \overbrace{\log_\phi (n+1)}^{\text{dominating term}}$$

$$n = O(\lg n)$$

How do we maintain property #2?

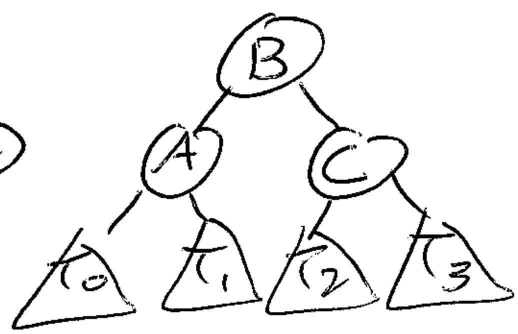
Key: rebalance operation



$$A < B < C$$

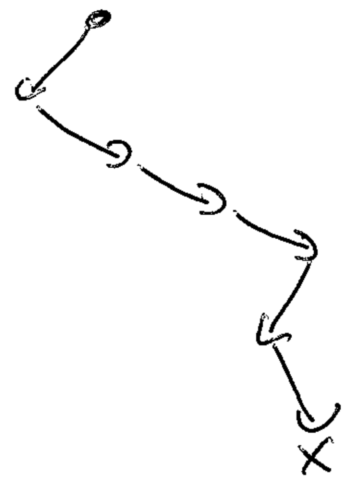
$$\text{all}(T_0) < A < \text{all}(T_1) < B < \text{all}(T_2) < C < \text{all}(T_3)$$

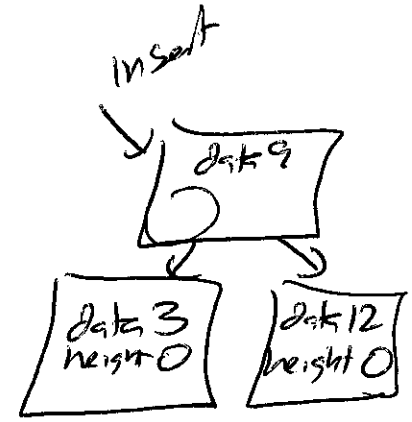
$\Longrightarrow$



Insert

① Recursive BST insert



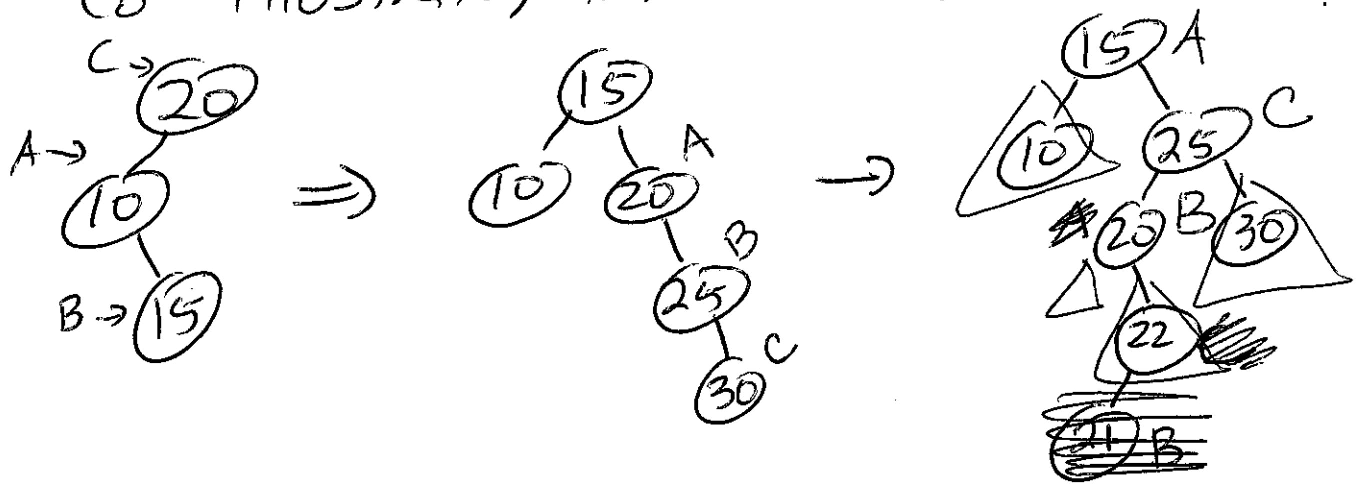


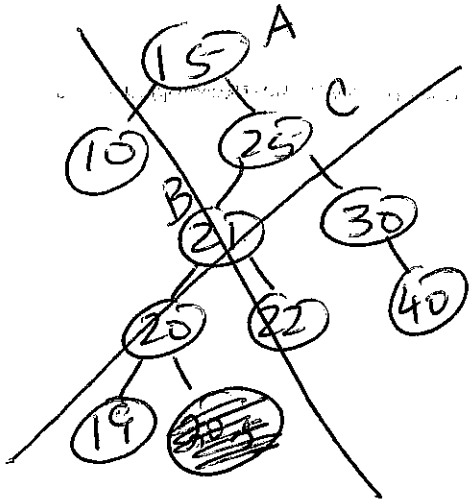
② Before you return ptr to root of updated tree,

look at L, R heights
if $|diff| > 1$, then call
rebalance on that node → ②b Update root node's height.

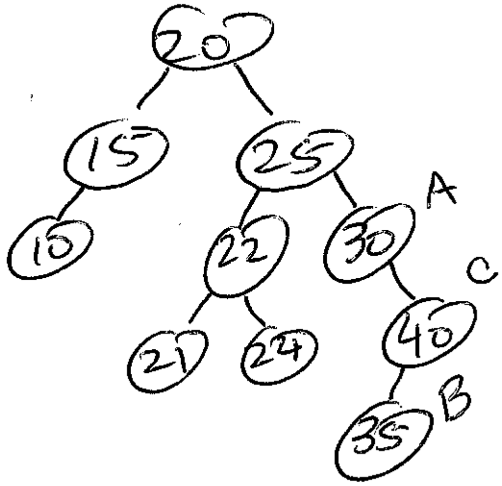
③ Return updated root

To illustrate, trace through examples!

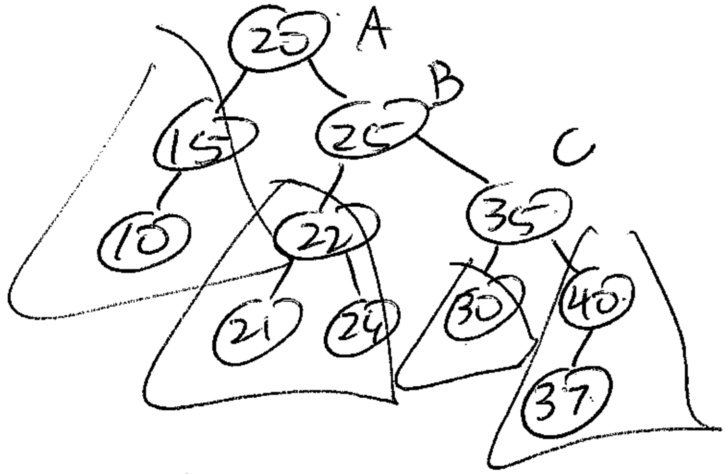



15
A
10
25
C
B
21
30
20
22
40
19

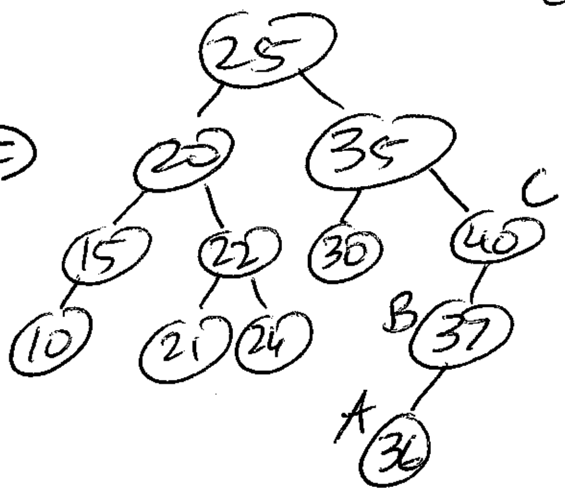
⇒


20
15
25
10
22
30
A
C
21
24
40
35
B

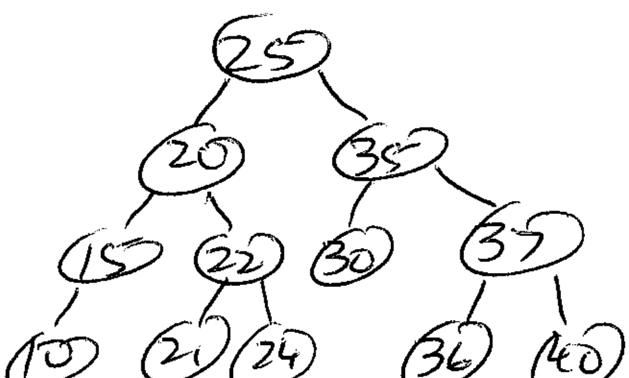
⇒


25
A
B
15
20
C
10
22
35
21
24
30
40
37

⇒


25
20
35
15
22
30
40
C
10
21
24
B
37
A
36

⇒


25
20
35
15
22
30
37
10
21
24
36
40

[redacted]