CSE 101 5-18-21

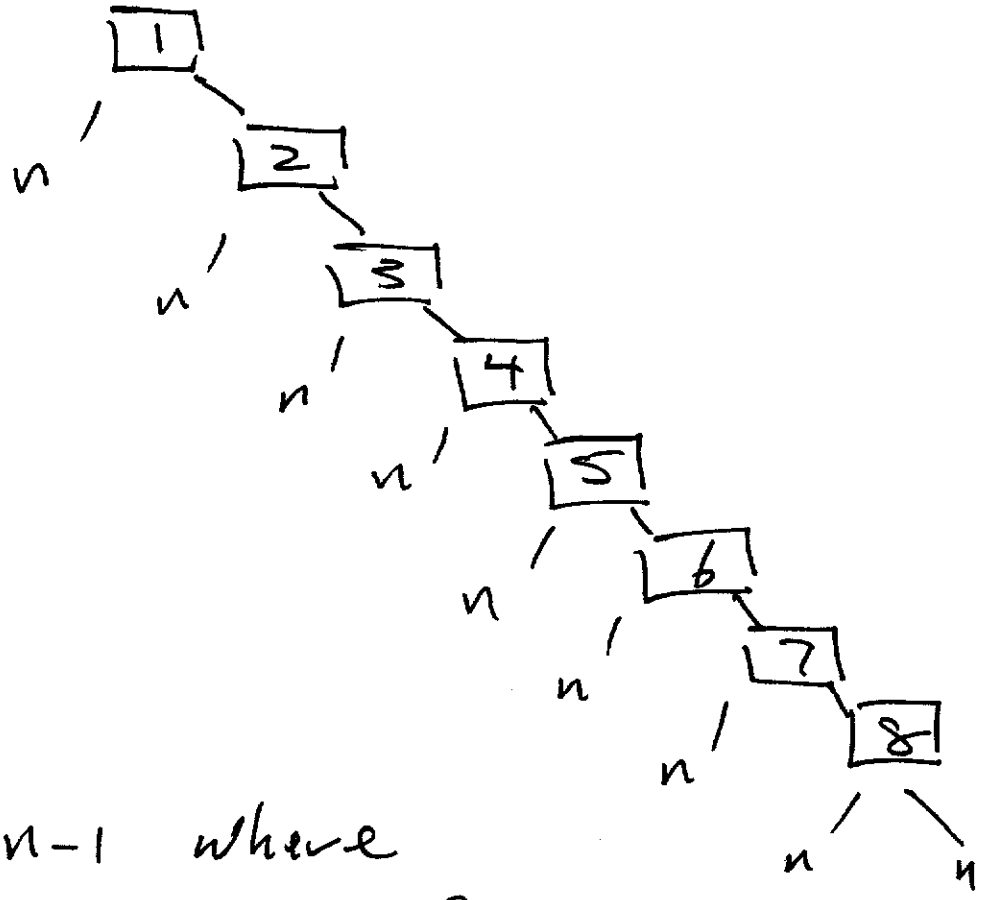
- quiz 4: today
- Pas: ext. 2 days

Runtime of BST Insert & Delete

both in time: $\Theta(\text{height}(T))$

A Problem arises if we insert keys in ascending order.

Ex. insert: 1, 2, 3, 4, 5, 6, 7, 8



height[T] = 7

in general: n-1 where

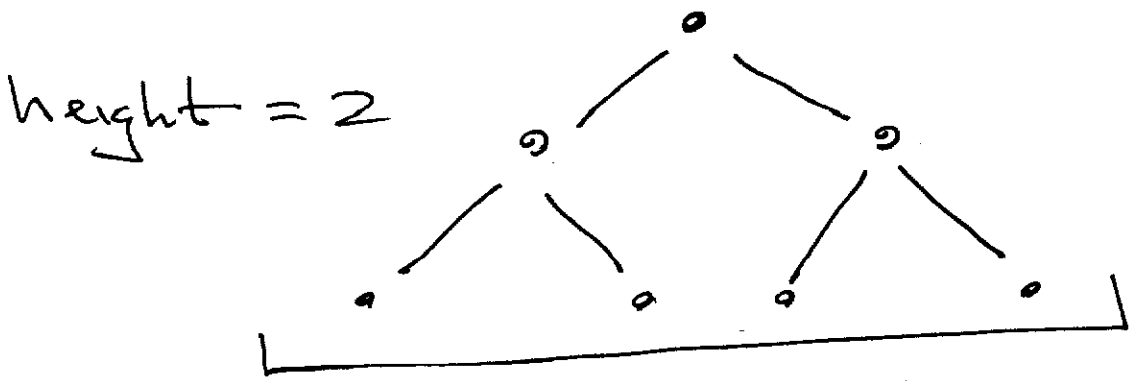
n = # of keys inserted.

One solution:

Randomly built BSTs.

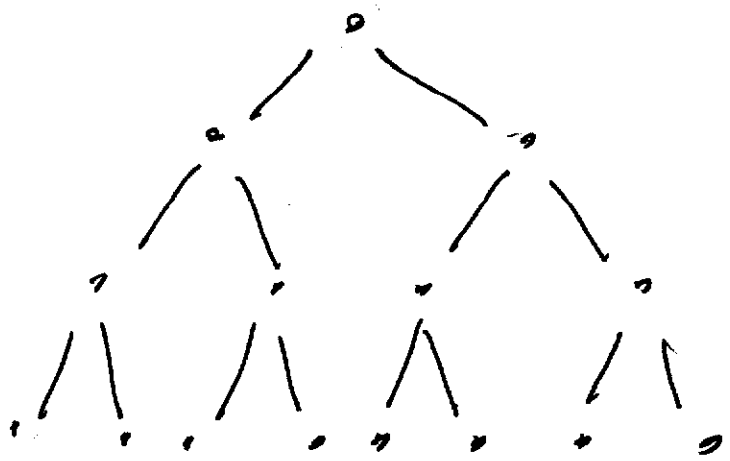
we have a high probability that a random insertion order will produce a good structure, i.e. many nodes with 2 children

best possible structure:



complete binary tree (CBT).

Ex. $n = 15$ height = 4



In general, a 'balanced' tree has

$$\text{height}(T) = \Theta(\log n)$$

Another solution (deterministic, not probabilistic):

Red Black Trees (RBT)

Goal: always produce a balanced tree.

chap. 13 in CLRS

A RBT is a BST, that in addition to the BST properties satisfies the RBT properties.

Convention:

The leaves in an RBT are considered to be the nil children of key bearing nodes.

RBT Properties

1.) Each node has a color: Red or Black

2.) root is Black

3.) Each leaf (i.e. nil child) is Black.

4.) Every Red node has 2 black children, one or both could be nil.

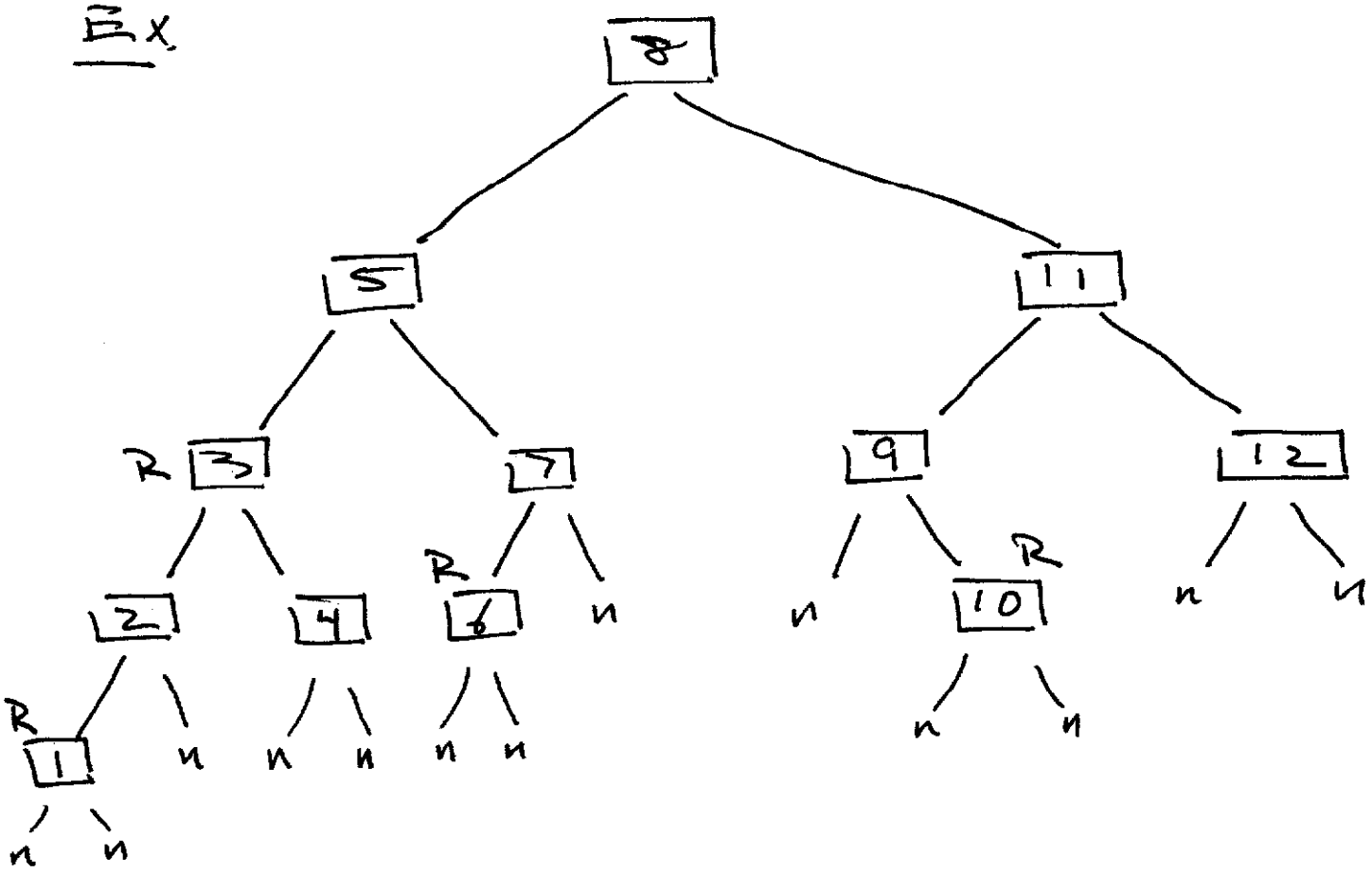
5.) for any node x, every descending Path from x to a leaf contains the same number of black nodes.

Def:

The Black Height of x, denoted bh(x), is the # of Black nodes any descending Path from x to a leaf (not counting x).

note we refer to height(x) as 'absolute height', to distinguish from black height.

Ex.



Nodes	Black Height
8	3
5, 11, 3	2
1, 2, 4, 6, 7, 9, 10, 12	1
all nil leaves	0

Note: $bh(x) = 0$ iff $height(x) = 0$ iff x is a leaf

Theorem

A RBT with n internal (key bearing) nodes and height h, satisfies

$$\boxed{h \leq 2 \lg(n+1)}$$

Proof

- Any Binary Tree satisfies

$$h \geq \lfloor \lg n \rfloor$$

Therefore a RBT satisfies

$$\lfloor \lg n \rfloor \leq h \leq 2 \lg(n+1)$$

$\lfloor \lg n \rfloor$: $\Omega(\log n)$; $2 \lg(n+1)$: $O(\log n)$

$\therefore$ height$(T) = \Theta(\log n)$

$\therefore$ Runtimes are $\Theta(\log n)$

- a Binary Tree satisfying

$$\text{height}(T) = \Theta(\log n)$$

is called a balanced tree.

- Insert and Delete for BSTs do not preserve RBT properties but they can be altered to do so, and still run in time

$$\Theta(\text{height}(T)) = \Theta(\log n)$$

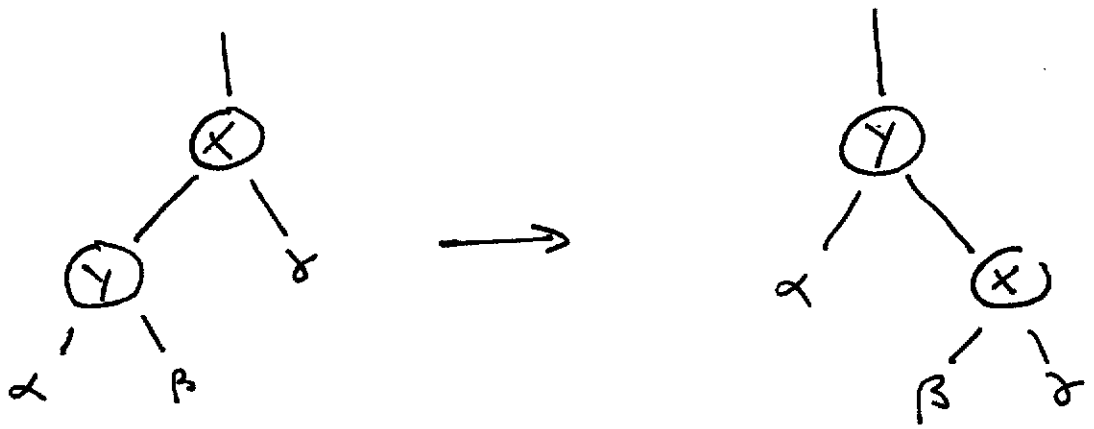
Sec. 13.2 Rotation

Strategy for Insert & Delete

- Do BST insert or delete
- Restore RBT properties if they are violated.

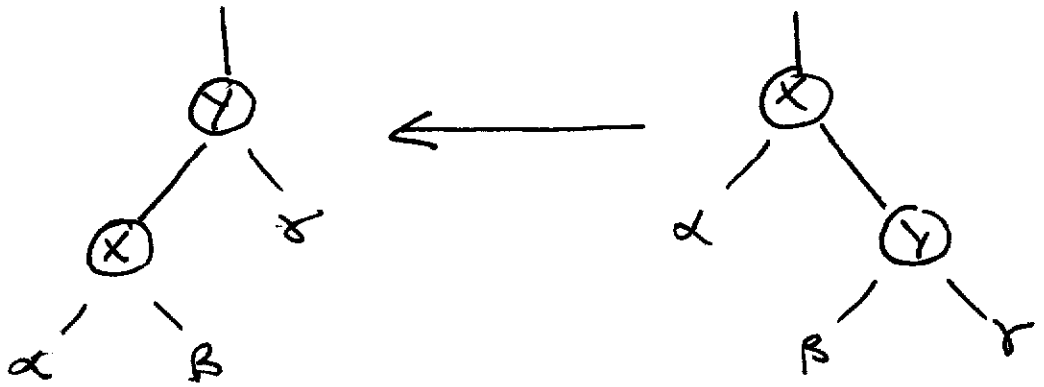
Two Rotate operations:

RightRotate(T, x)



α, β, γ are arbitrary subtrees

LeftRotate(T, x)



- Both operations preserve BST properties
- Both <u>do not</u> preserve RBT properties.

summarize RightRotate(T, x)

- change β :

$$x.left = y.right$$

$$y.right.Parent = x$$

- fix Parent
- fix x, y

} next time

Pseudo-code } next time