

CS2100

AVL Trees



This week + next:

- HW due today - given website error, may submit tomorrow.

- Next HW: → remove in a BST
 ^{next} remove in an AVL
 _{Tues} ↪ wed/thurs next week

↪ tomorrow will be a work day
(no lab)

- Today: office hours at 2pm
(noon - 2:30)

- Likely not be a lab next
 Tuesday - lecture instead

- Midterm 2: April 16

Last time:

Binary Search trees.

What's left - runtimes?

find:

$O(n)$

①

②

③

④

⑤

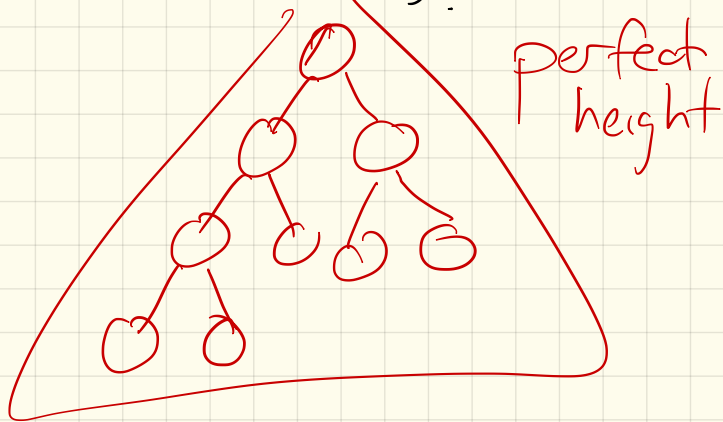
insert:

remove:

Alternative: $O(\text{height}(T))$

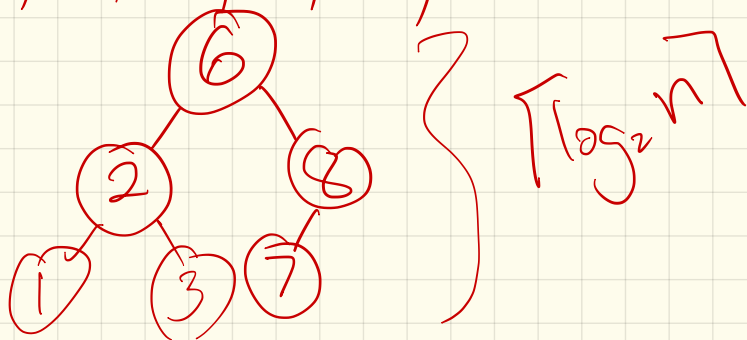
Today: How to make better?

Well, why were heaps $O(\log n)$ instead of $O(n)$?



Goal: How can we balance?

Ex: 1, 3, 8, 7, 6, 2



Balanced BSTs

Many kinds:

- Red-black trees : $1.4 \lceil \log_2 n \rceil$

- Splay trees :

- AVL trees : $2 \lceil \log_2 n \rceil$

⋮

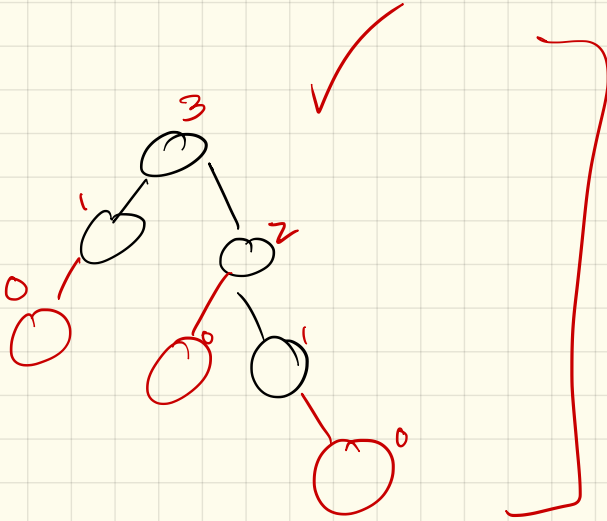
Goal of them all :

$O(\log_2 n)$

AVL trees :

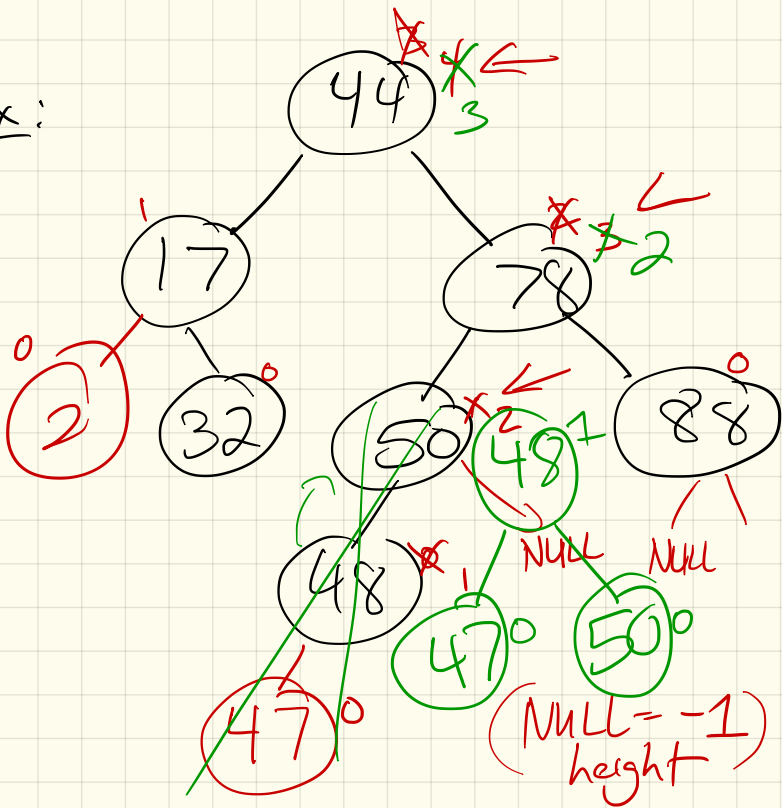
Height balance property :

For every node x in T ,
the heights of its children
differ at most 1.



$$\Rightarrow \text{max height} \leq 2 \lceil \log_2 n \rceil$$

Ex:



Now: how can we mess this up?

insert(2) \leftarrow OK

Ex: insert(47) \leftarrow NO!

Fix it! How? rotate (or pivot) 48 up

Consider the lowest node which fails the Height-Balance property.

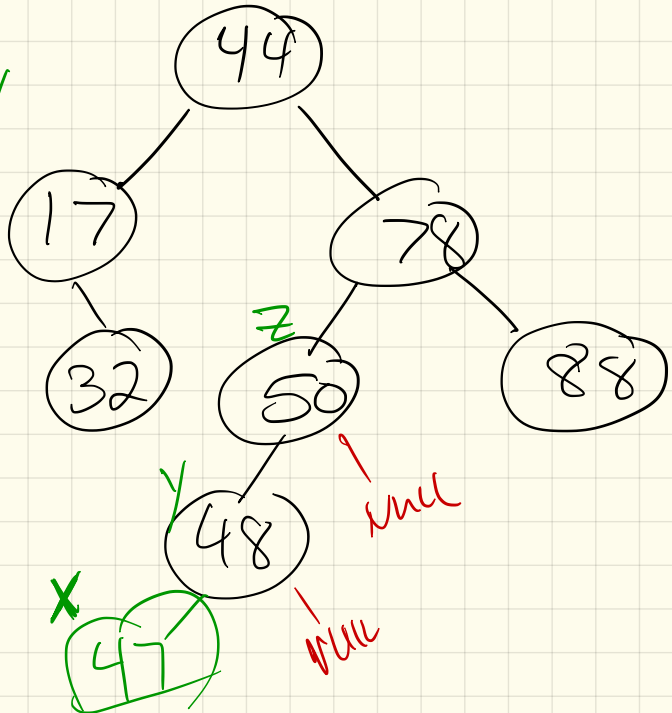
↳ call this z

Let y be child w/ bigger height.

Let x be grandchild w/ larger height.

Fix!

move y up + balance

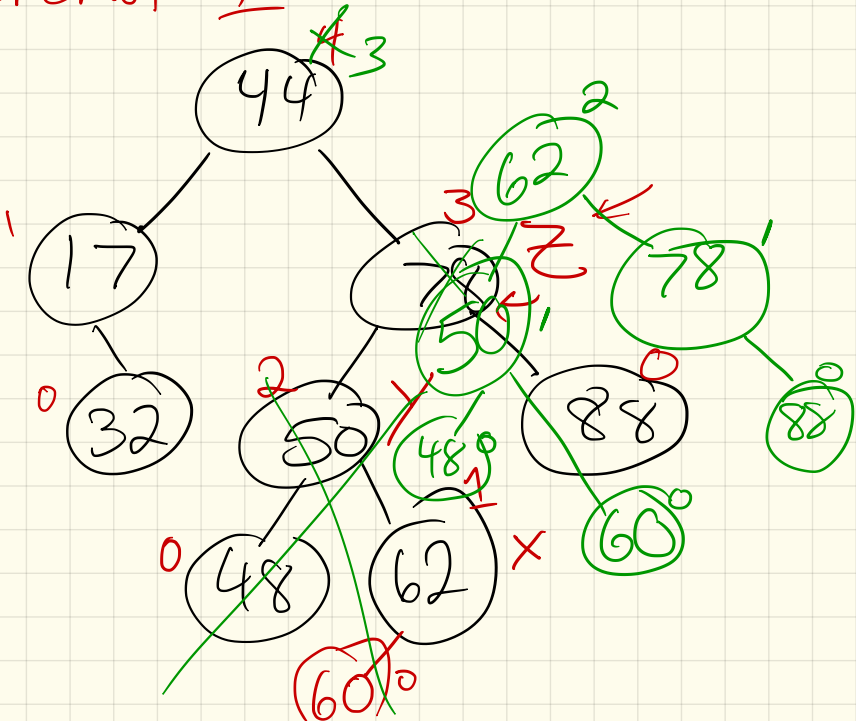


Another: insert(60) below
 Consider the lowest node
 which fails the Height-Balance
 property.
 ↳ call this **Z**

Let **Y** be child w/ bigger height.

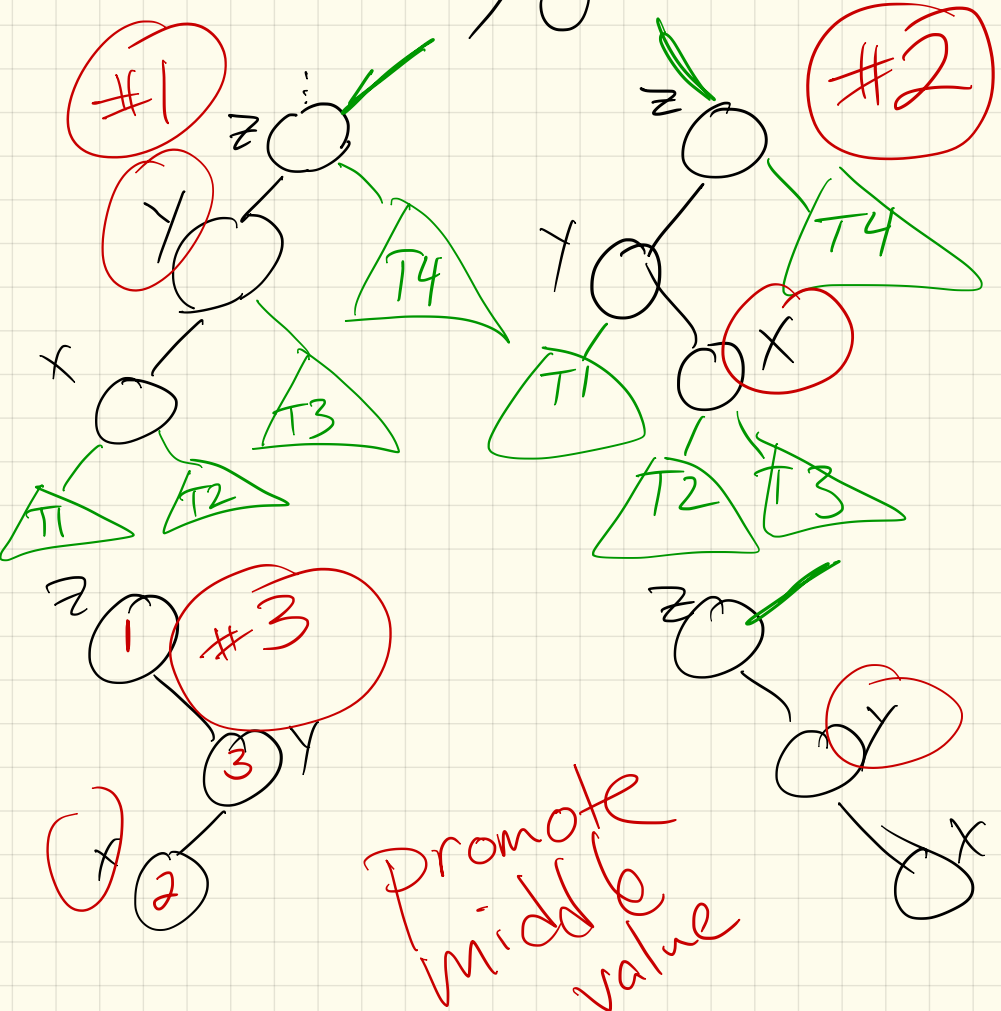
Let **X** be grandchild w/ larger height.
 ↳ promote **X**

Fix!

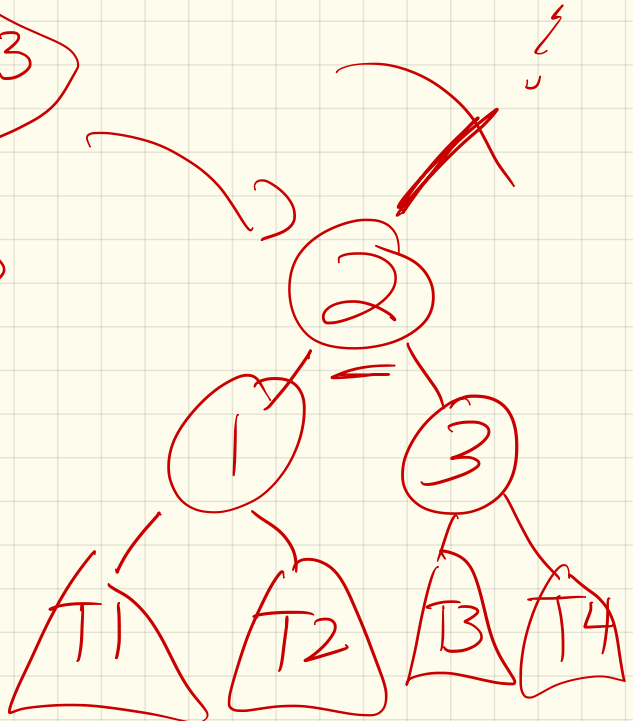
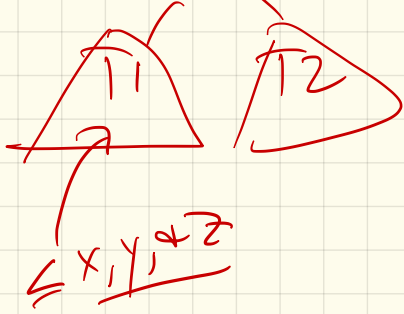
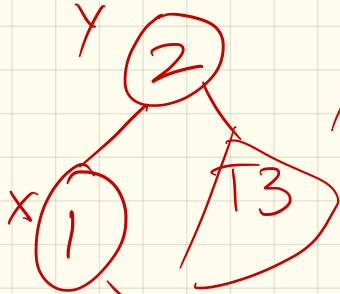
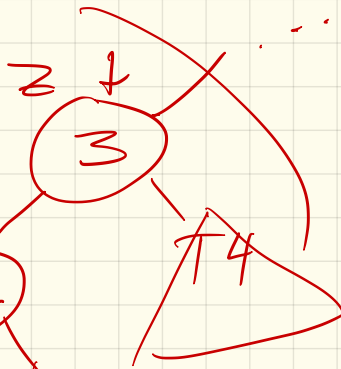


So: algorithm to insert:

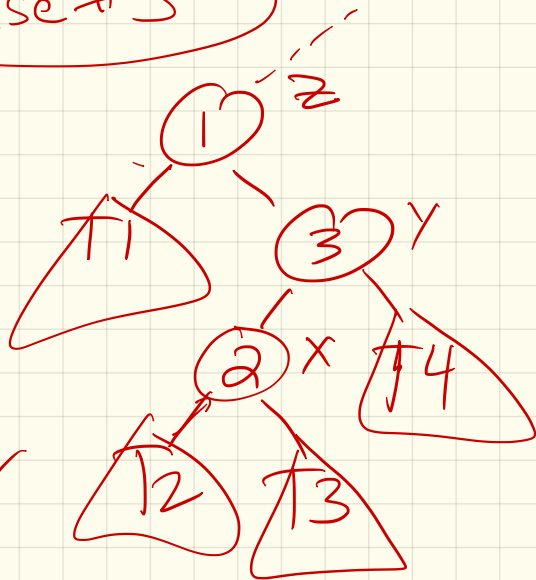
- First run BST insert
- Then find lowest unbalanced node Z & deeper child/grand child.



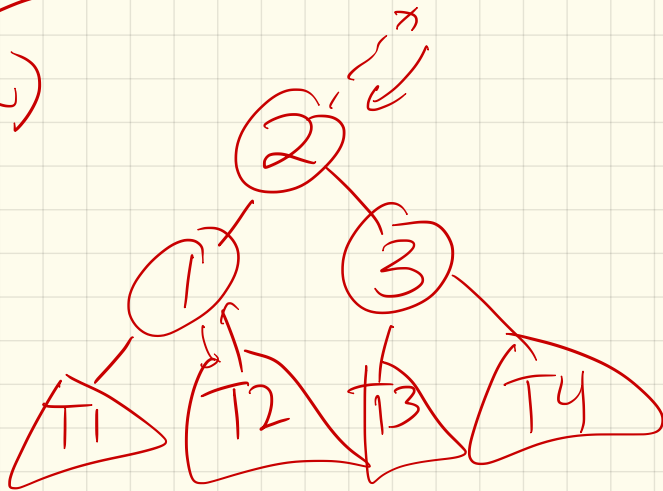
#1



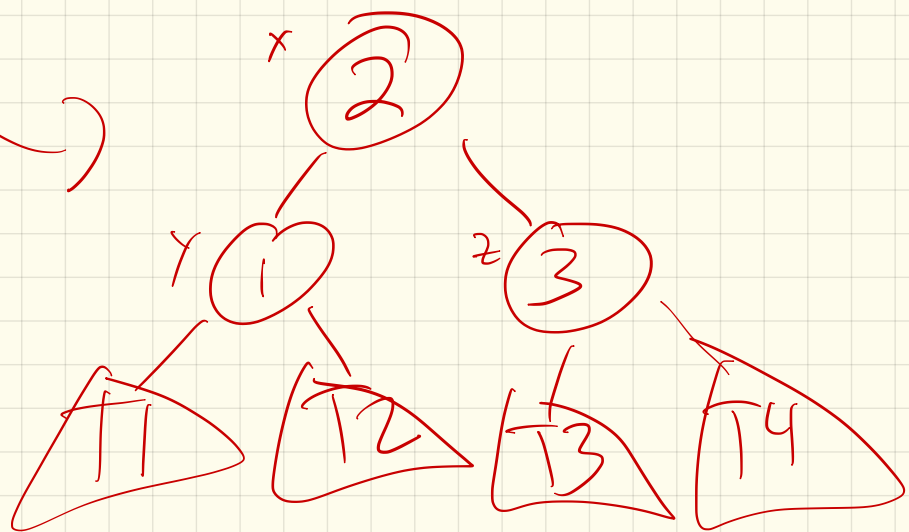
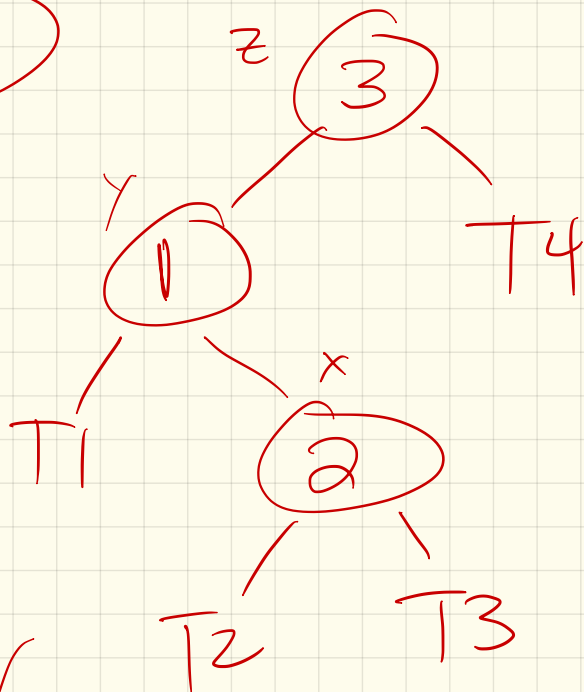
Case #3



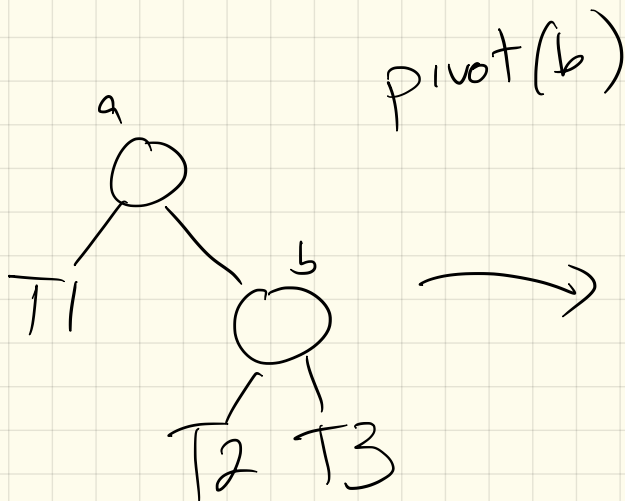
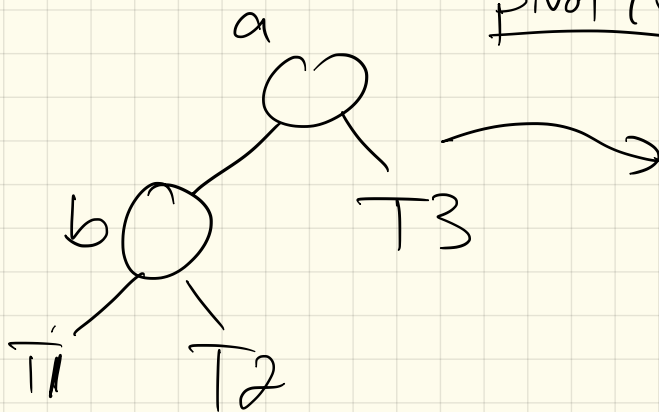
fix : make x the root



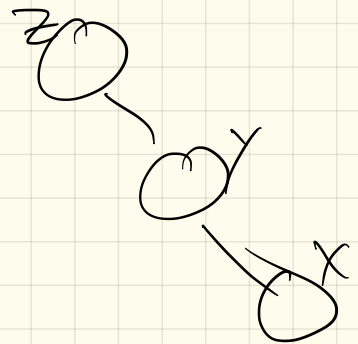
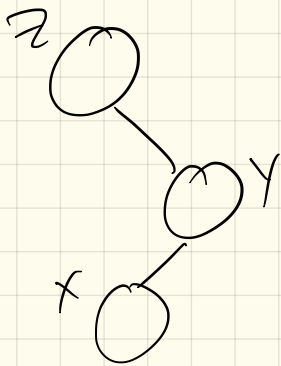
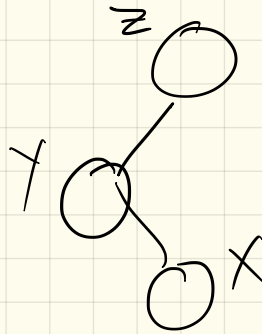
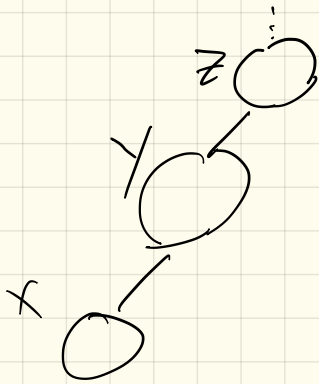
#2



Key operation : pivot
pivot(b)



Then: implement w/ pivot!



Bigger example:

Insert: 1, 2, 3, 4, 5, 6, 7