CS2100

Treaps (cont)

Recap:

- HW due Thursday
- Lab tomorrow
  (also one next week)
- Review Friday, test in 1 week

Treaps: a new binary tree structure
(Aragon + Seidel, '89)

Goal: Each node will contain a value (like a BST) and a priority (like a heap).

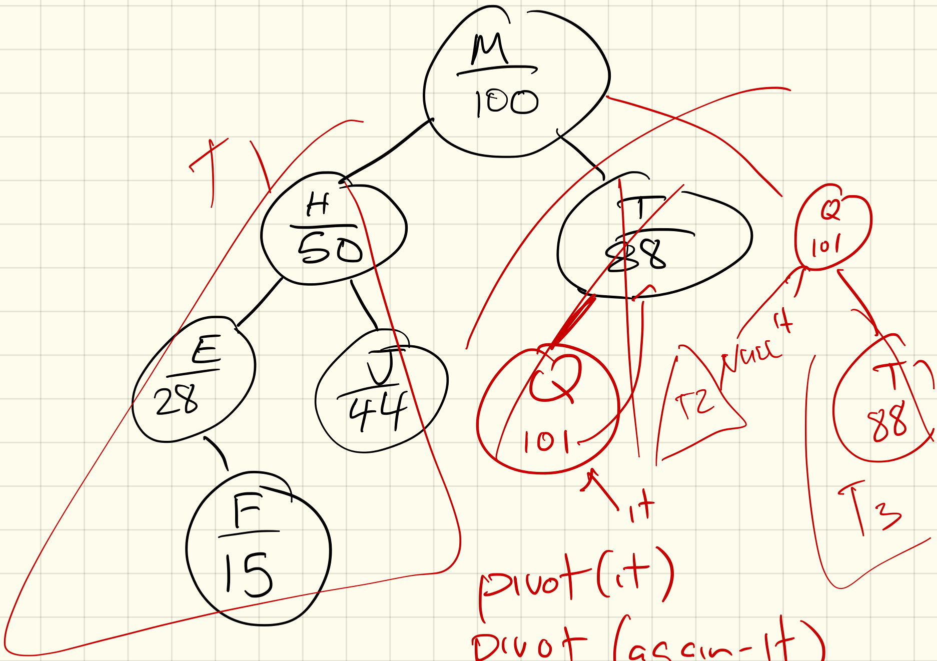
- BST over values
- heap over priorities

Ex: Suppose values are names and priorities are integers.

Both can be "sorted":

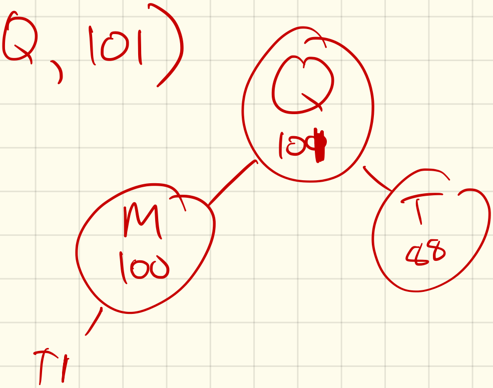
- values/names have alphabetical order
- integers (obviously)

Example



pivot(it)
pivot (again-it)

Insert : (Q, 101)

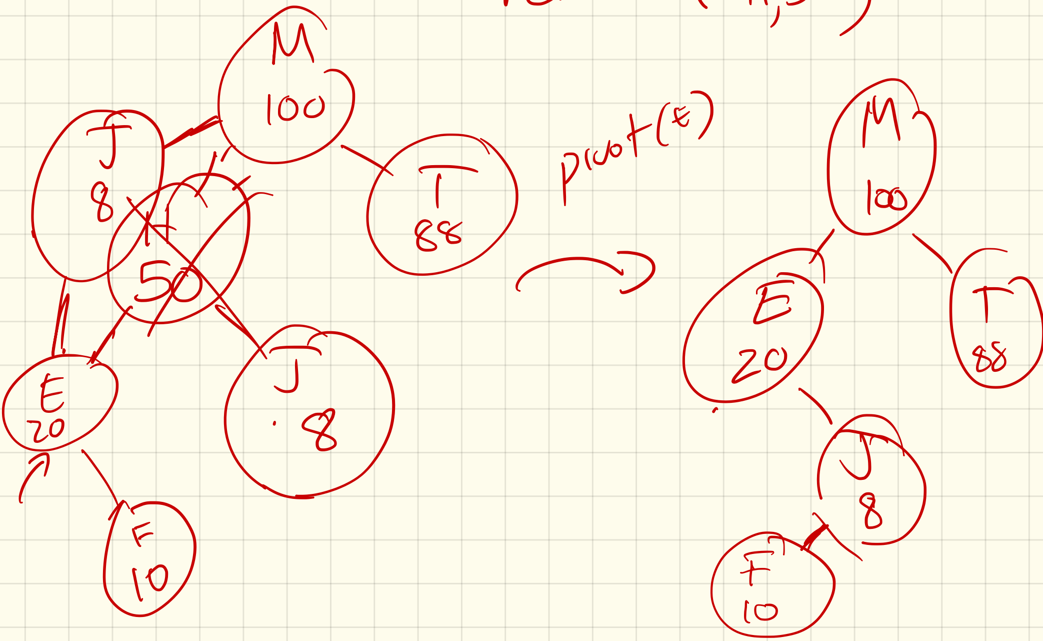


Removing:

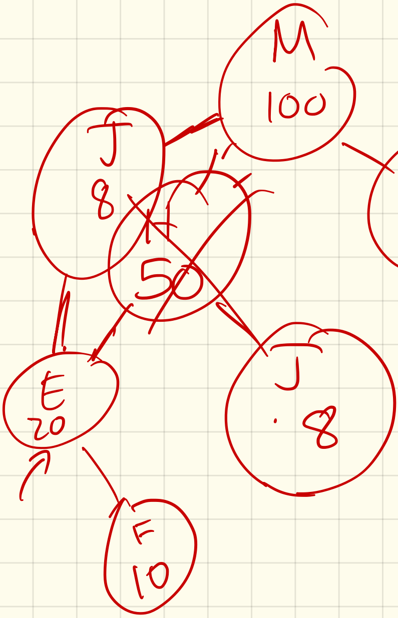
- Do BST remove
- Fix priorities

Note: pivot ~~up~~ or down

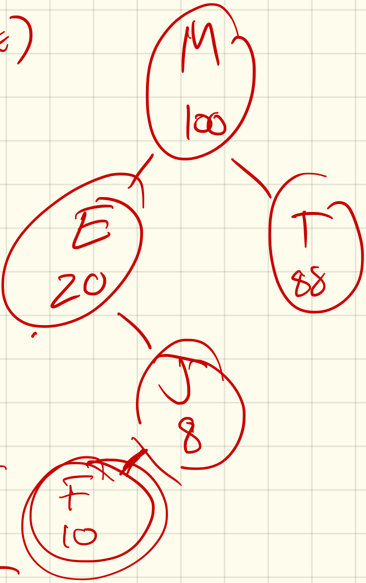
remove(H,50)



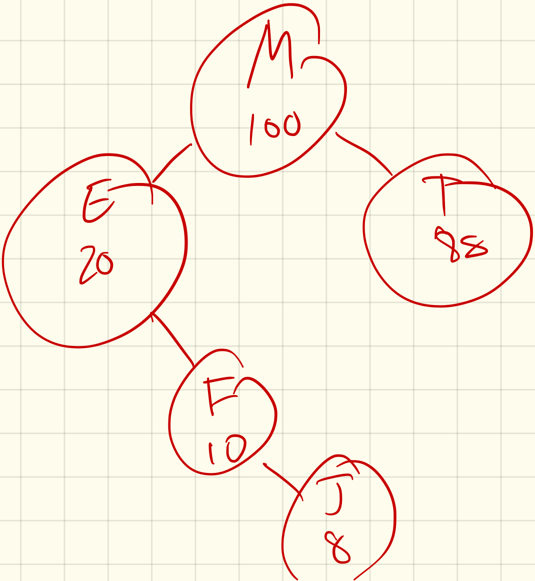
remove (H,50)



pivot(E)



pivot F



Implementation:

- Inherit from binary search tree

  _data: values (letters)

  _aux: priorities (ints)

- use BST's insert/remove, & binary tree's pivot to fix

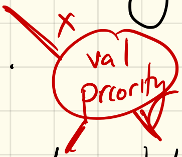
Avoid: AVL's get & set height

Note: Treaps are unique!

Given a set of values/keys, order of insertion is irrelevant.

pf: Consider one valid treap w/ set of values + keys.

Consider x, a node.



If we change x's height:

means child/parent swaps – violate priority

If we change x's order:

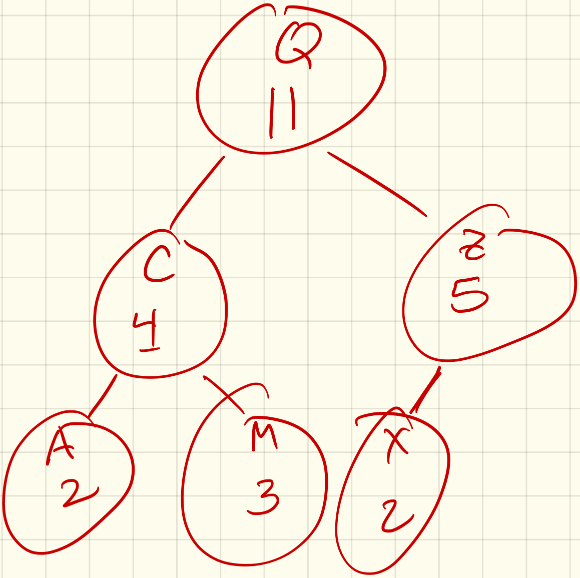
violates BST

Ex: Draw heap with:

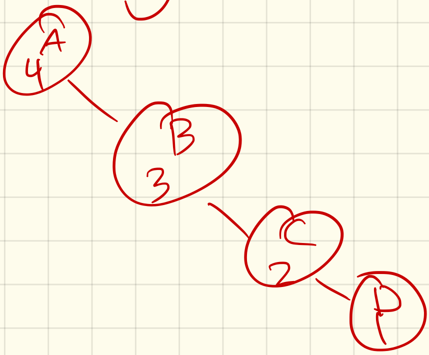
L L

(A, 2), ~~(C, 4)~~, ~~(Q, 11)~~,
(X, 2), (Z, 5), (M, 3)

R R L



Worst case height:



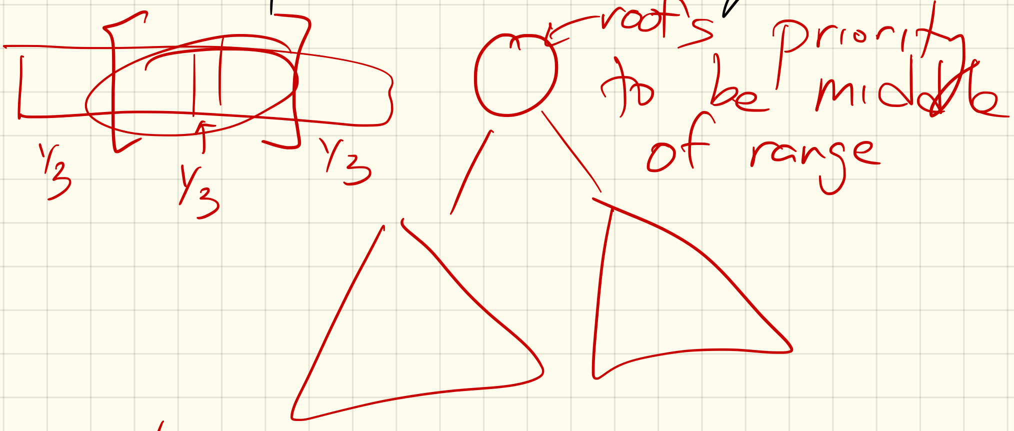
Randomized treaps : Balanced BST

Alternative to AVL trees.

Given a value to insert, give it a random priority.

Thm: Expected height of the treap will be $O(\log n)$.

Why? remember quicksort:



root's priority to be middle of range

w/ prob. $\frac{1}{3}$, get "good enough" root

From here:

These will be on written HW, due towards end of semester