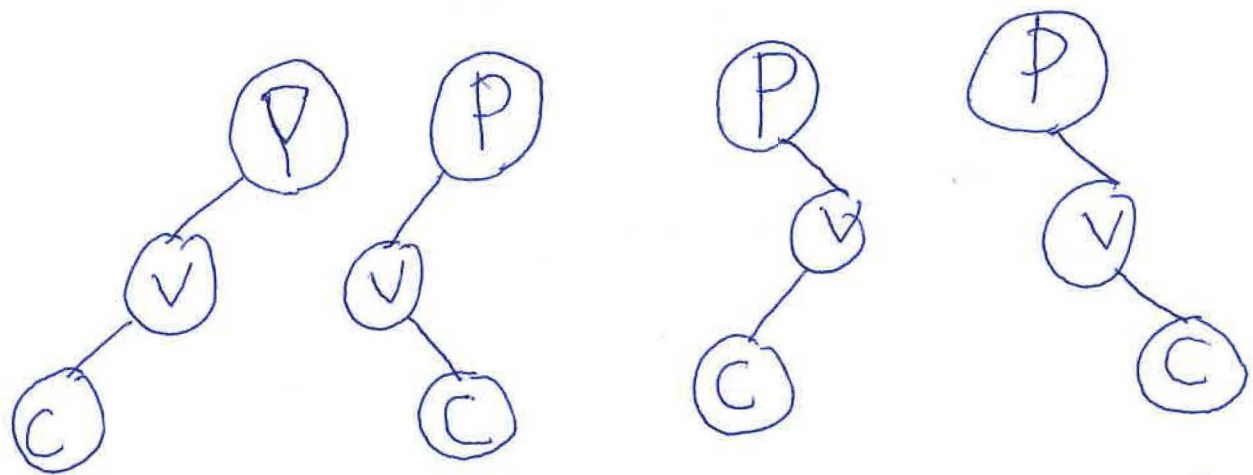


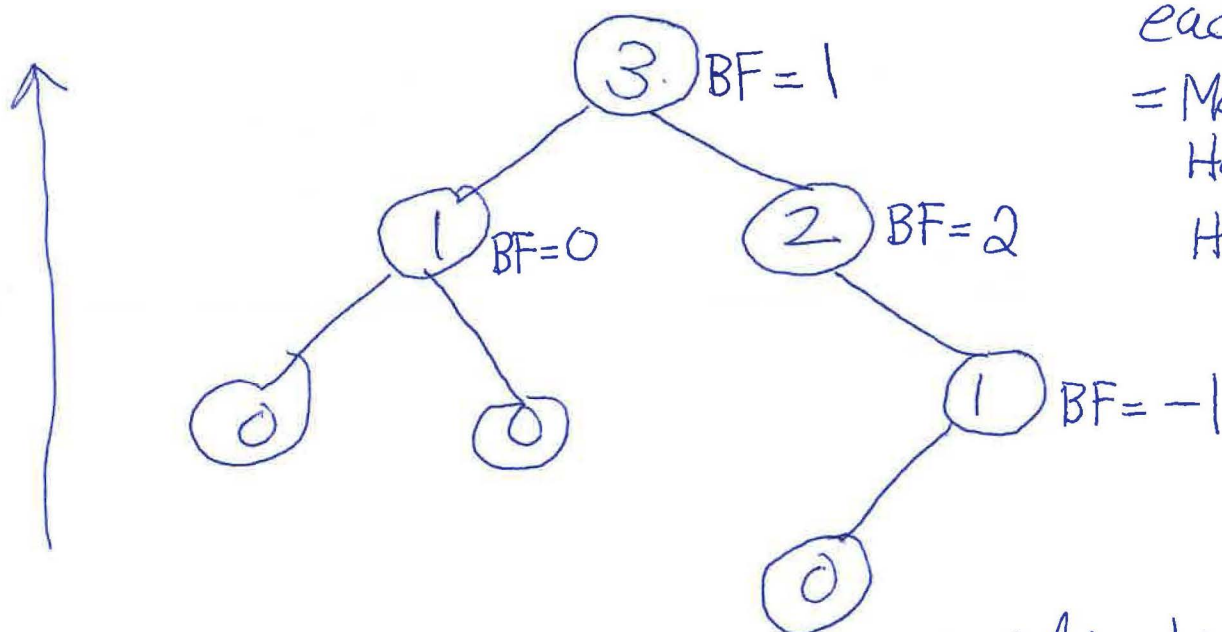
Deletion Case 2 (By Pass)



take care of all 4 cases
& set the Up link of C.

Height & Balance Factor

Height
inside
each node
= Max of
Height of left
Height of Right
+ 1



BF = Height of Right child - Height of Left child
Height of NULL is -1

Data Structure Summary

Stack: push and pop to solve a problem

Queue: add and remove to solve a problem

Linked List: add anywhere, delete from anywhere, search sequentially (dynamic memory)

Search Tree: add fast, delete fast, **search fast**, sort fast (if balanced, add/delete/search is $O(\log N)$)

Heap Tree: add fast, delete fast, **keeping it sorted** (is always balanced and can be in static memory)

Can you pick the right data structure for a problem?

PQ Heap Summary

You are using slots 0 through count-1

Remove from slot 0

Move the last element to slot 0 and re-heapify (trickle down)

Add to the very end

Trickle up

Trickle Down: $LC = 2I + 1$ $RC = 2I + 2$

Trickle Up: $I = (LC - 1)/2$ or $(RC - 2)/2$ LC is odd RC is even

