CS301 LinXiLi

Right-Rotate:

```
x = y.left
y.left = x.right
if x.right ≠ T.nil
    x.right.p = y
x.p = y.p
if y.p = T.nil
    T.root = x
elif y == y.p.right
    y.p.right = x
else y.p.left = x
x.right = y
y.p = x
```

13-3:

a. If a ~~subt~~ AVL tree exists, than it must have two subtrees of it's root node. one is h-1 (h is the height of the tree) Another one is h-2. or h-1

so $T(h) \geq T(h-1) + T(h-2)$), this implies that (T(h) is the minimum number of heighth tree) ~~T(h) = T(h)~~

$$T(h) \geq T(h-1) + T(h-2).$$

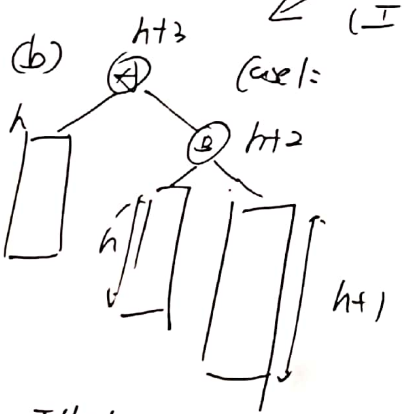
$$F_h = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^h - \left(\frac{1-\sqrt{5}}{2}\right)^h}{\sqrt{5}}$$

$$T(h) \geq F_h = \frac{\cancel{\ }}{\sqrt{5}} \cancel{\ } (X)^h$$

$$h = O(\lg)^n.$$

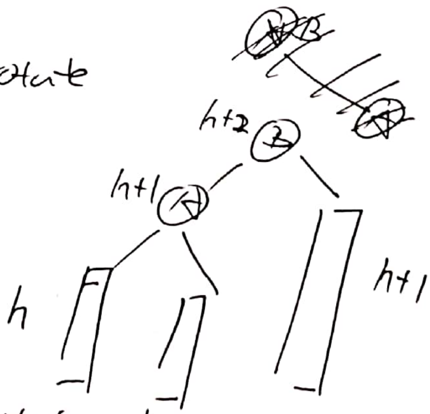
(I searched this (b) on google to get some inspirations)

(b)



case1:

Left rotate

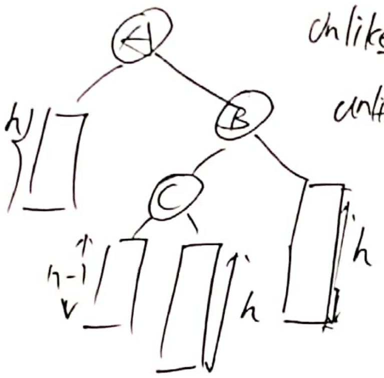


this would be balanced

I think we can simply left rotate the whole tree if it's case1.

Case2:

If there's some subtree in the right subtree (or "bigger" subtree) of A
It will be case2.

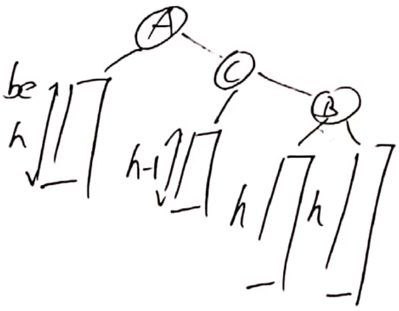


~~unlike case1, h is~~

unlike ~~case1.~~ h is ~~different~~

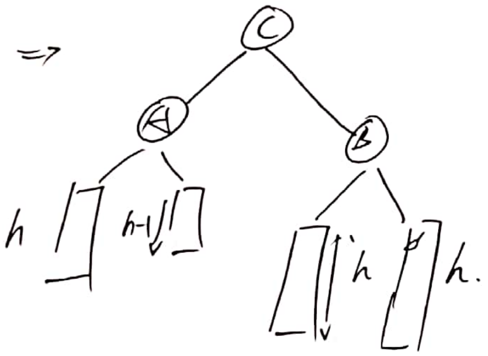
→ we first rotate the right subtree of A.

It turns out to be



, then we can left rotate the whole tree

⇒



so the Procedure's Pseudocode would be
Let x denote the root node of the whole tree

```
Balance(x)
  if (height(left[x]) - height(right[x])) ≤ 1
     return x.
  else {
       if h(left(x)) > h(right(x))
          y = left(x)
          if  height(left(y)) > height(right(y))
              Left-Rotate(y)
          else:
              return RIGHT-ROTATE(x)
       else    y = right(x)
          if height(left(y)) > height(right(y))
              right-rotate(y)
          return Left-rotate(x)
```

Insert(X, Z)

(c) If X = NIL:

    return Z ~~m~~ (the root)

~~else~~ if Z < X:

    y = Insert(~~&~~ X.left, Z) ⇒

        X.left = y.

~~else~~ else:

    y = Insert(X.right, Z) ⇒

        X.right = y

X = Balance(X)

return(X)

(d). since the height of AVL tree is O(logn). so insertion and update will Perform O(logn) time to insert.

And since the balance will only affect certain Part of the tree, so It's going to be O(1)