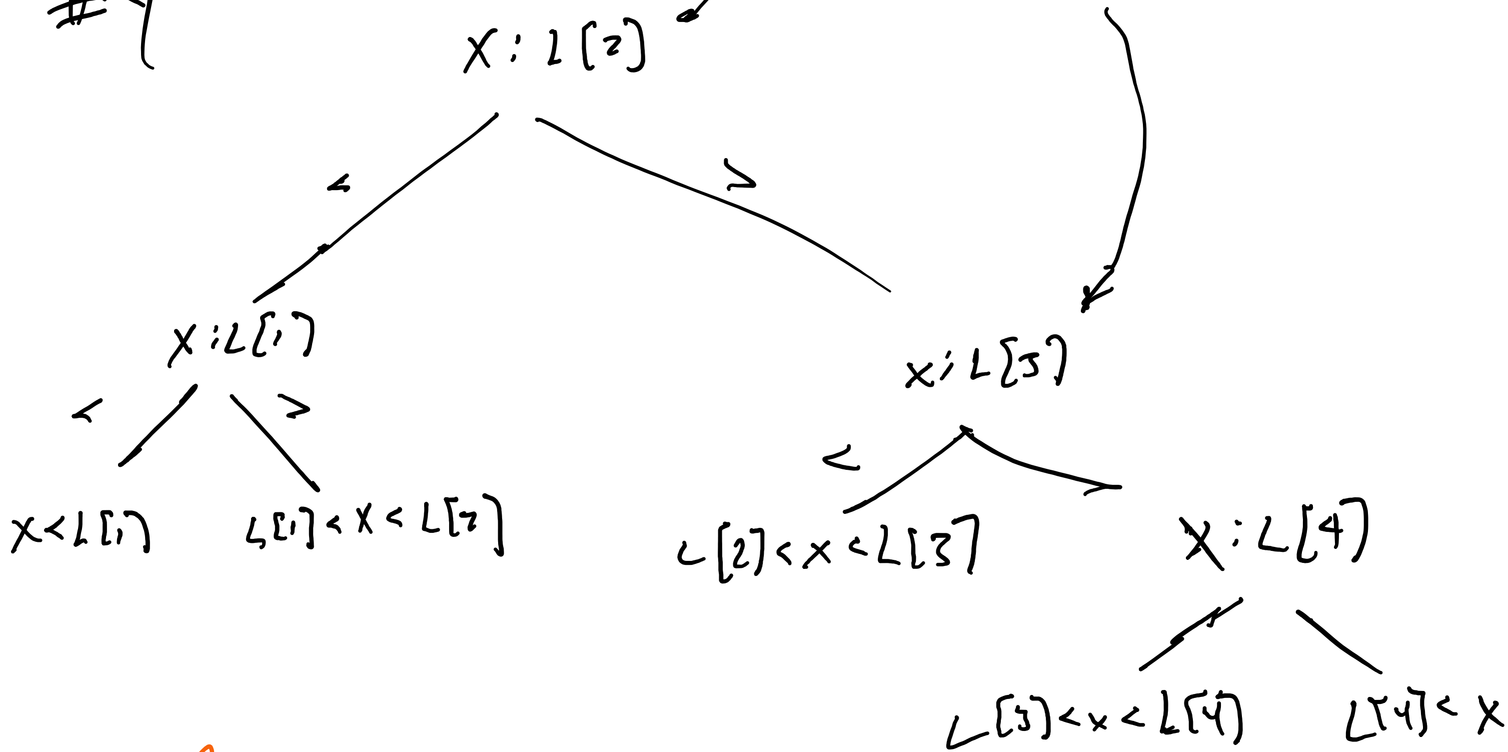
§ 6.3 # 4, 10, 17

4

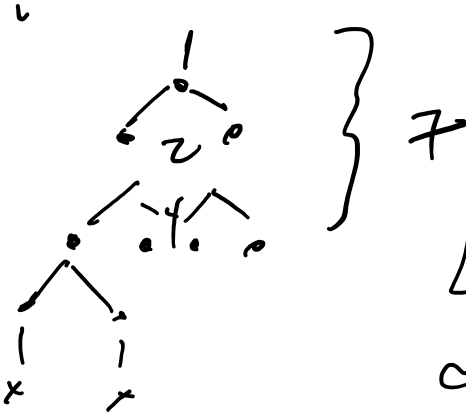
Top of left is author's algorithm



depth: 3

Worst case: 3 comparisons

#10 a. If we pack the n items into a bushy tree, we'll get our minimum # of comparisons.



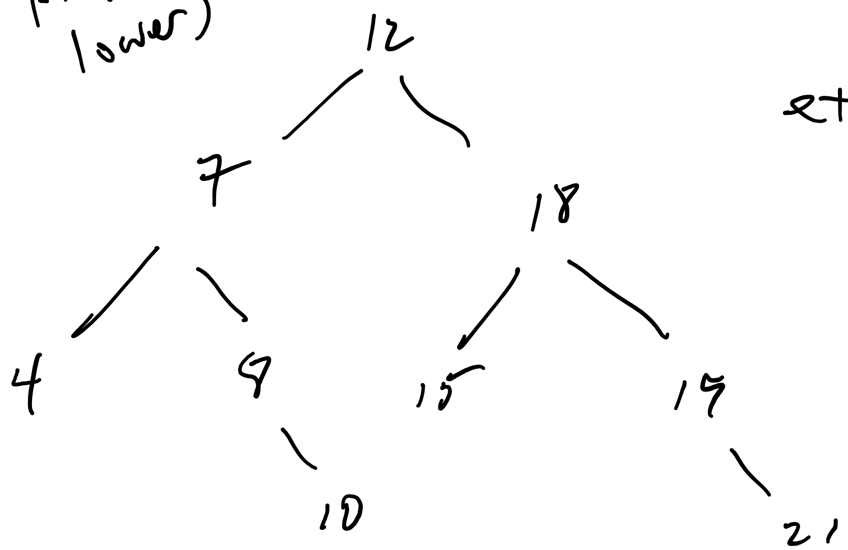
$\lfloor \log 9 \rfloor + 1 = 3 + 1 = \boxed{4}$ comparisons

leaves at depth 4 → 4 comparisons.

b. Make a binary search tree by processing data "middle first":

4, 7, 8, 10, 12, 15, 18, 19, 21

(stop at lower) ② ① ③

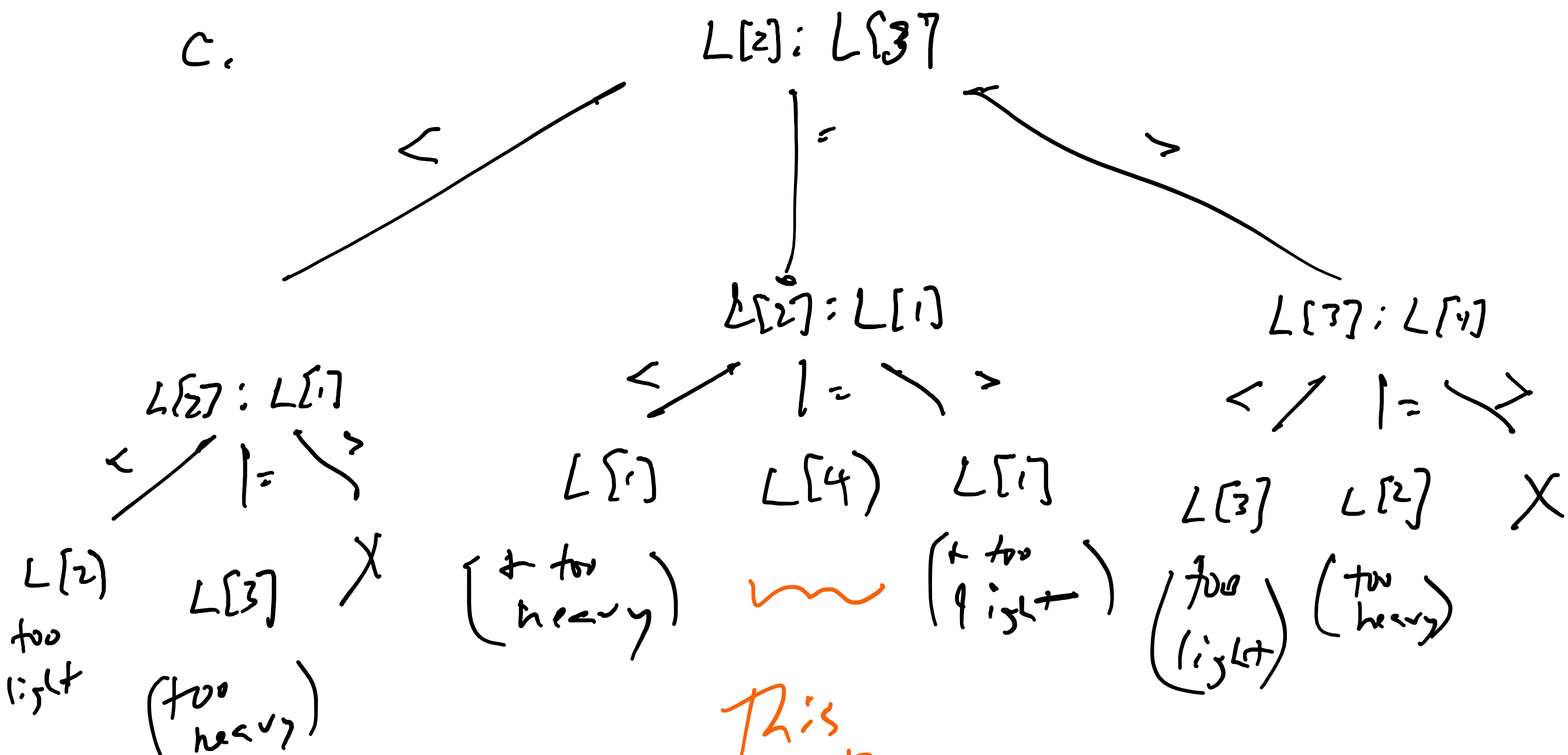


etc.

#17 a. Leaves - four - the counterfeit coin!

b. 2 - depth of a ternary tree with at least 4 leaves

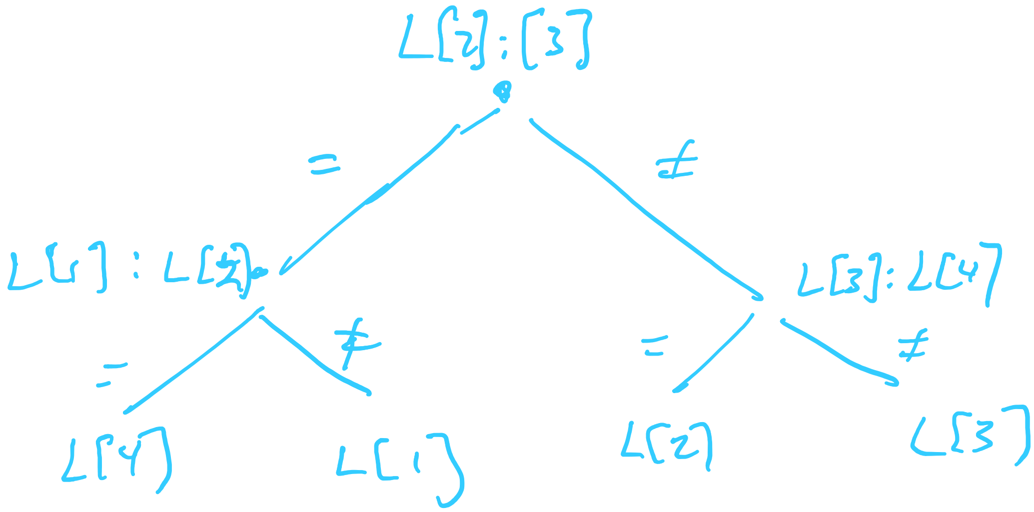
c.



This is the only case we can't determine the "direction" of the problem - heavy or light.

Would require one more weighing!

We're squeezing as much info as we can out in this process. But if we don't want to squeeze so hard, we can go binary!


L[2] : [3]
=
≠
L[1] : L[2]
L[3] : L[4]
=
≠
=
≠
L[4]
L[1]
L[2]
L[3]