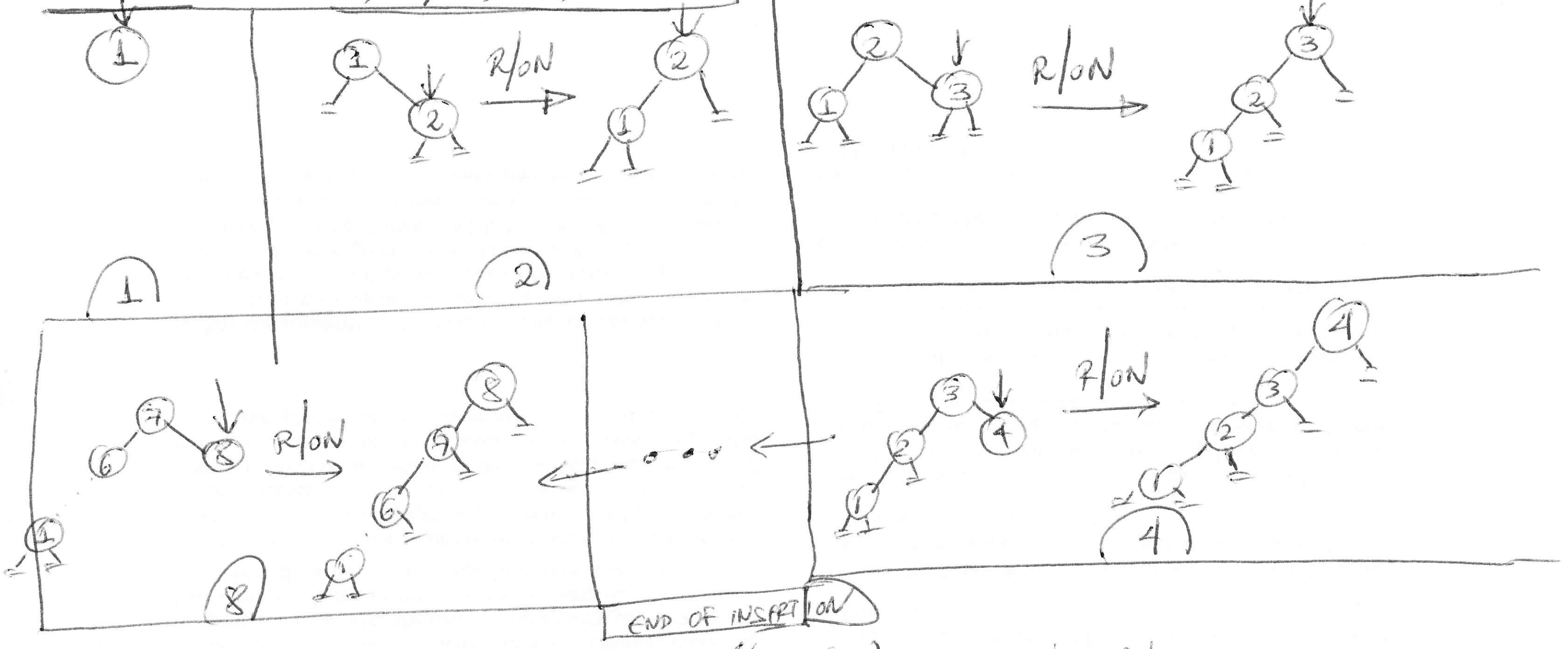
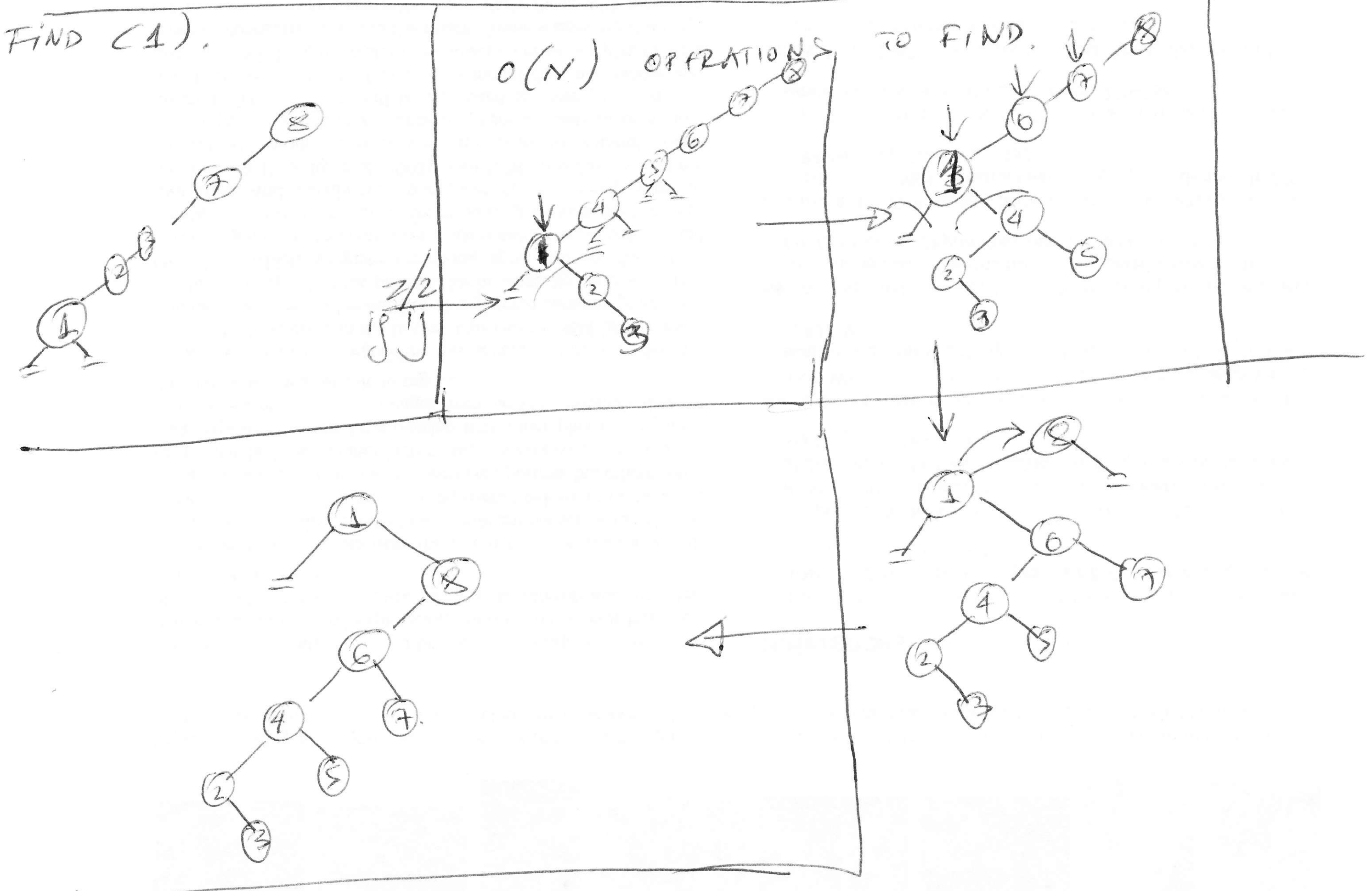
INSERT : 1, 2, 3, ..., 8



COST PER OPERATION : O(1). // N(=8) OPERATIONS

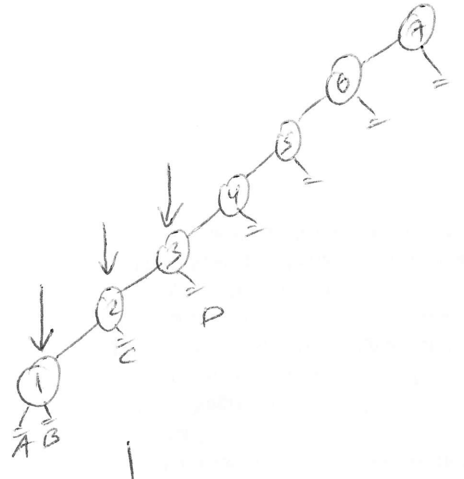
FIND (1).



O(N) OPERATIONS TO FIND.

# of rotations? O(N) ⟹ TOTAL O(N).

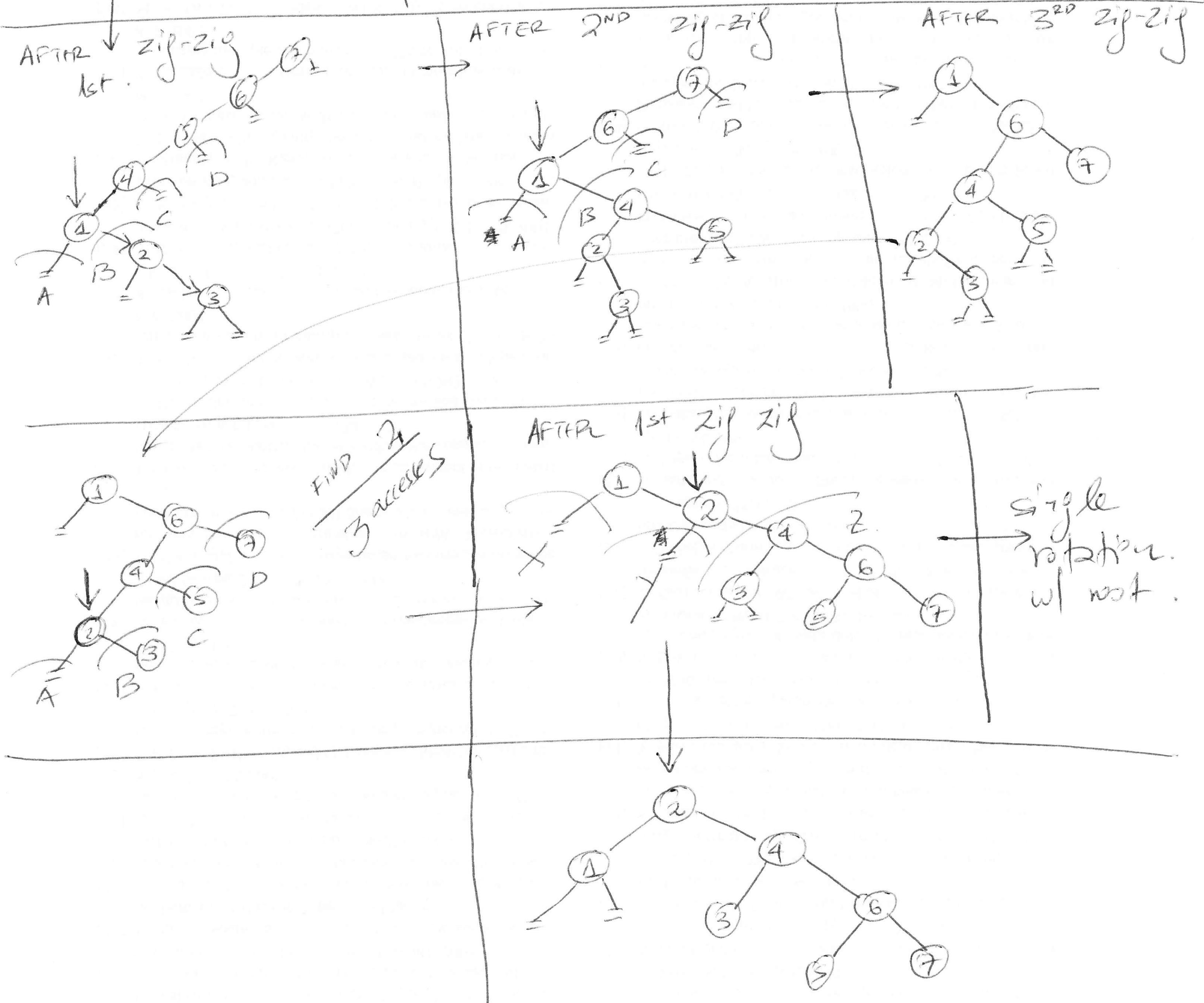
AMORTIZED COST? $\overset{\text{FIND}}{O(N)} + \overset{\text{BUILD}}{O(N)} = \frac{O(N)}{N+1 \text{ op/os}}$ → O(1)



ACCESS/FIND 1. N=7.

O(N) TO FIND

THEN Z/Z options.


AFTER 1st zig-zig

AFTER 2ND zig-zig

AFTER 3RD zig-zig

FIND 2

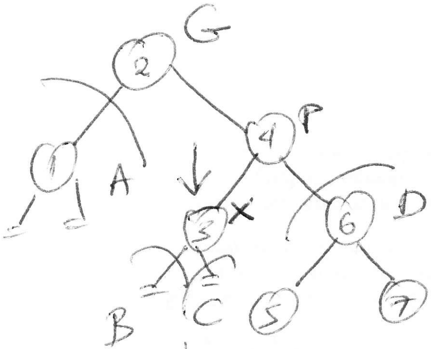
3 accesses

AFTER 1st zig zig

single rotation w/ root.

N=7

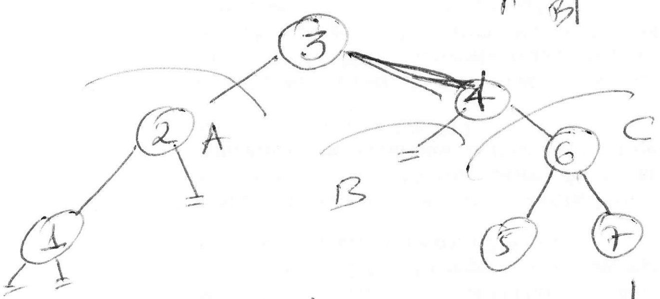
FIND 3



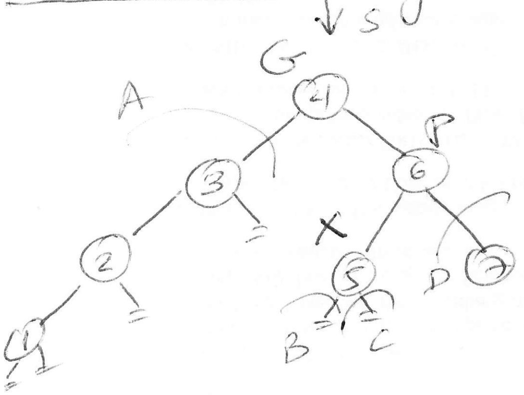
double rotation. (Right-Left)

FIND 4

ALMOST B/CED.

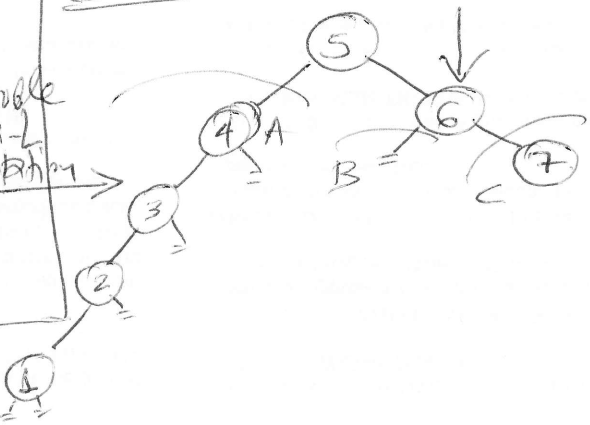


single rotation w/ root.



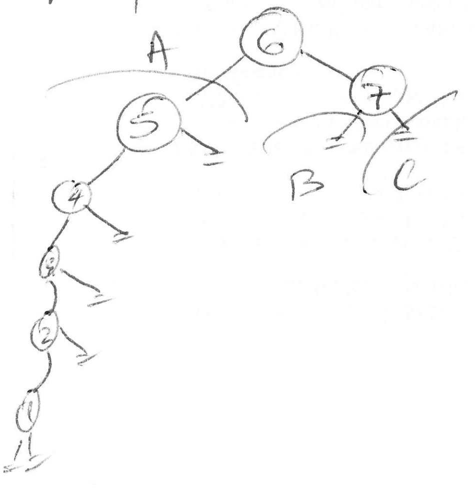
double R-L rotation

FIND 5



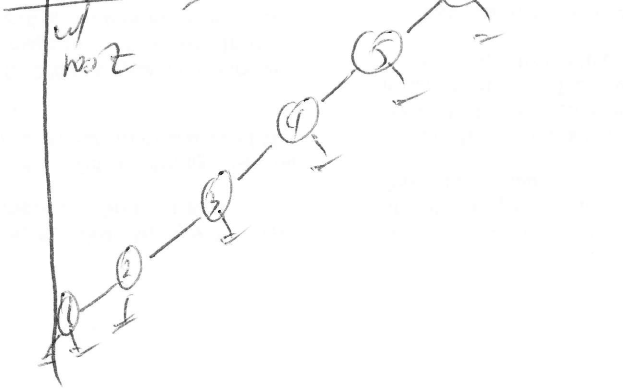
FIND 6

S/r w/ root

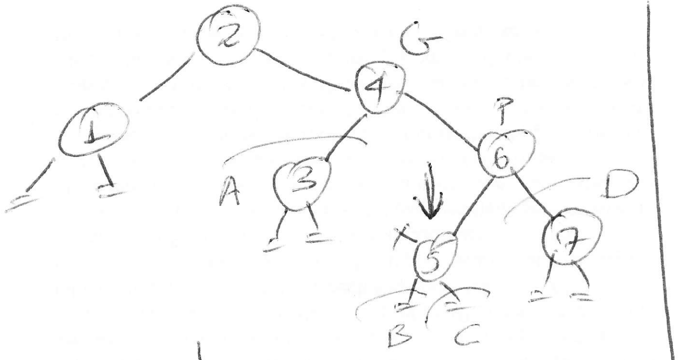


FIND 7

S/r w/ root



EX/LE of DELETION.
START W/

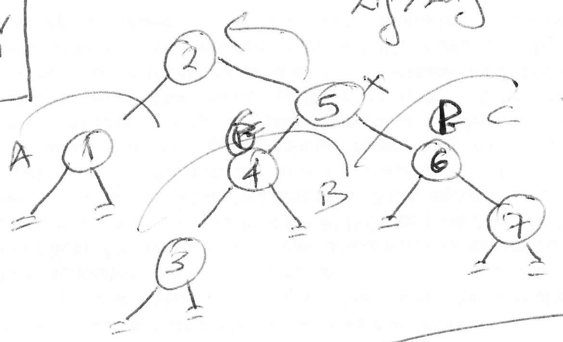


DELETE 5.

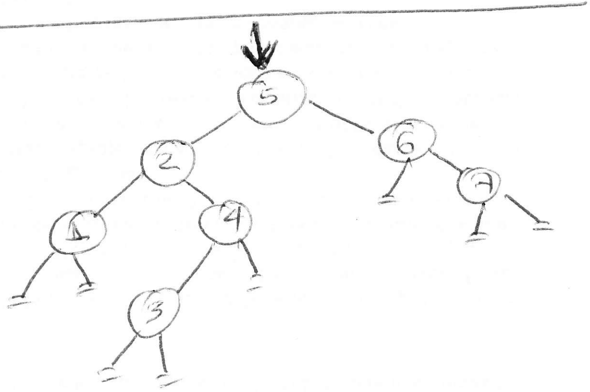
R/L double rotation
zig/zag

ACCESS 5
AND SPLAY

(1)



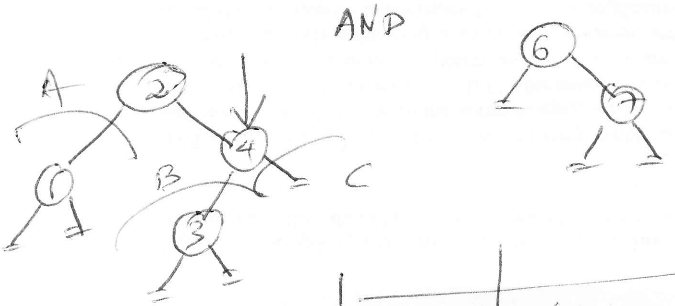
s/R



(2)

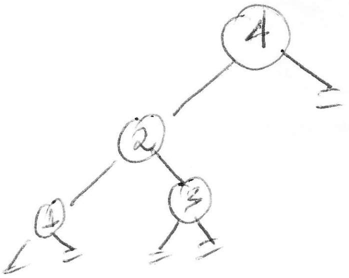
DELETE 5
TWO SUBTREES.

AND



(3)

ACCESS LARGEST
EL/NT OF RIGHT SUBTREE
BRING IT TO ROOT.

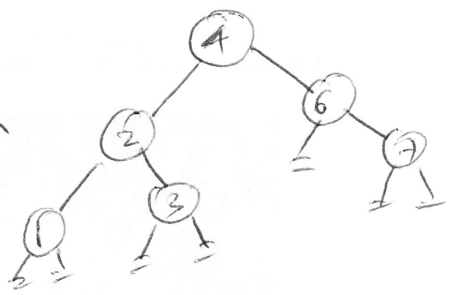


NO RIGHT
CHILD.

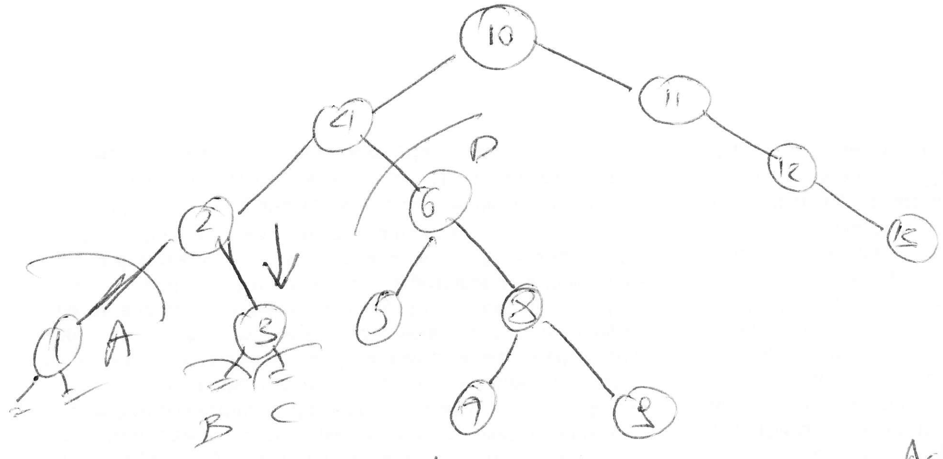
(4)

LEFT

DONE.

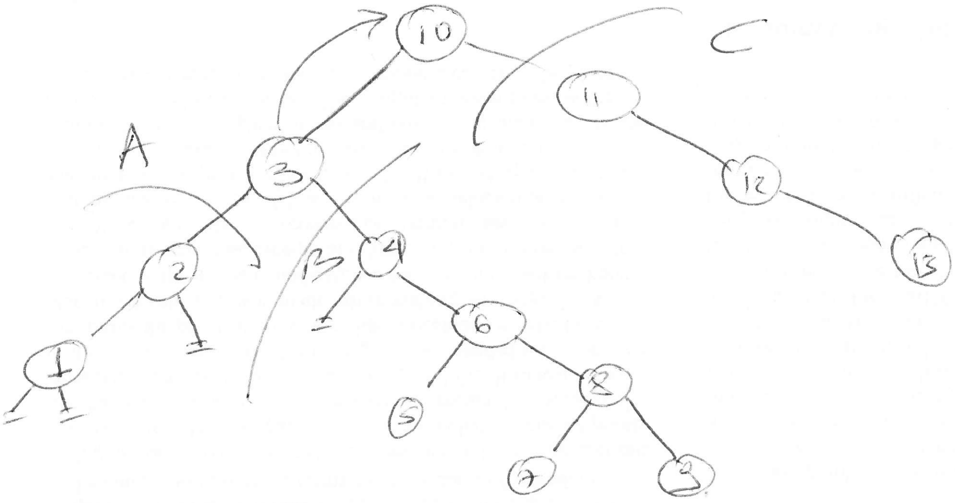


4.27

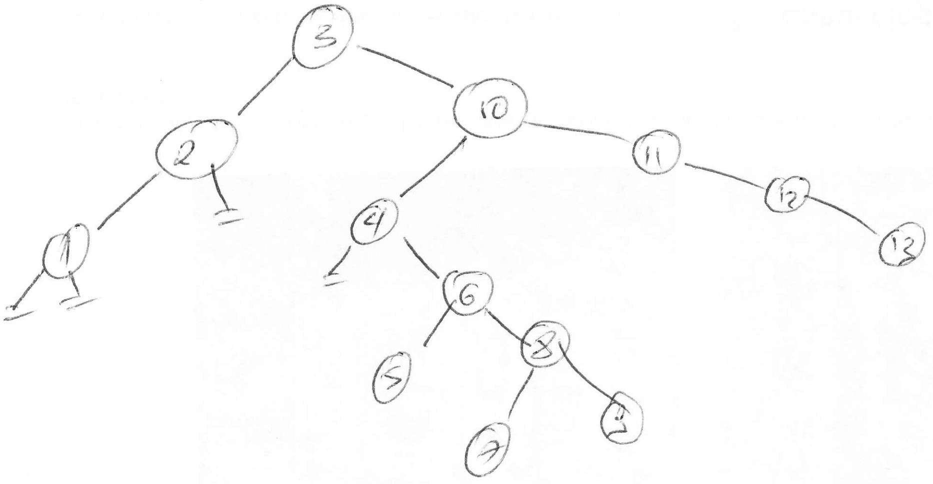


ACCESS 3

L/R D. ROTATION



S/RON.


3
2
10
11
12
4
13
1
6
5
8
7
9