Delete

Balance factor of Imbalanced Node X

> 1

(L) Left heavy

Balance factor of Left child of X

>= 0

(L)

LL

Action

Right Rotate around X.

Balance factor of ~~Right~~ Left child of X

< 0

(R)

LR

Action

1) Left rotate around X. left

2) Right rotate around X.

Balance factor of Imbalanced Node X

< -1

(R) Right heavy

Balance factor of Right child of X

<= 0

(R)

RR

Action

Left Rotate around X

Balance factor of Right child of X

> 0

(L)

RL

Action

1) Right rotate around X. right

2) Left Rotate around X.

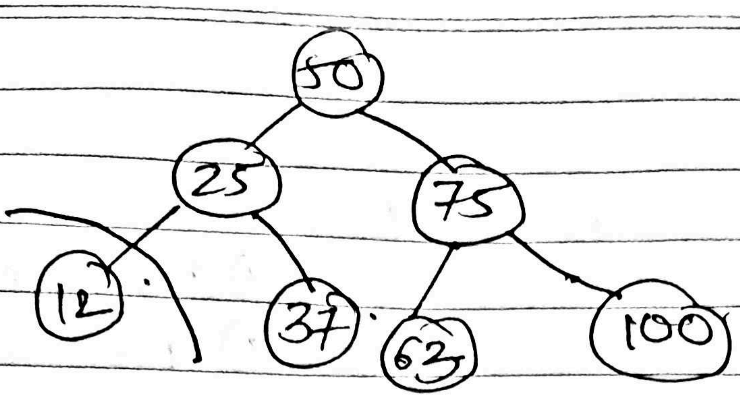
After deleting a node, check the nodes on the path from deletion point to the root.
If balance factor of a node is invalid ~~(=~~ (< -1 or > 1), then check which child & which subtree is heavier (using B.F.) and accordingly apply rotations.

Difference between Insert & Delete

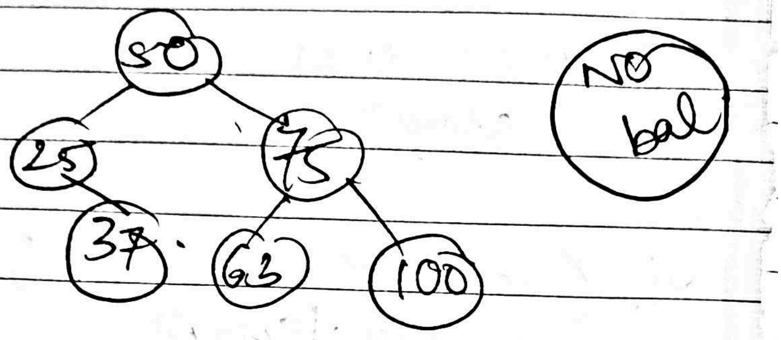
After Insertion, max. 2 rotations are required to balance the tree.

After deletion, the imbalance may propogate upwards, which then requires more than 2 rotations.

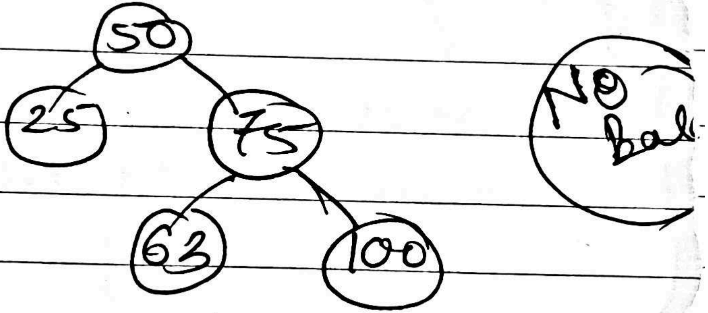
1



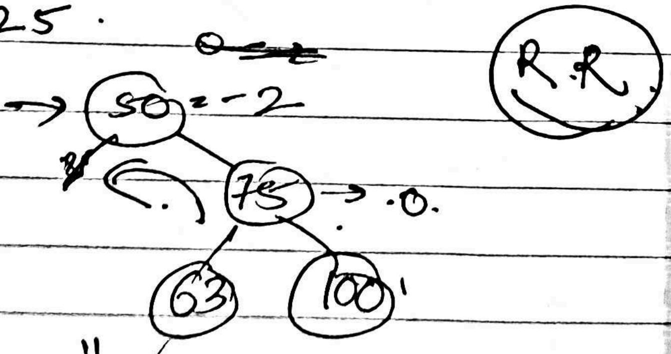
Delete: 12.



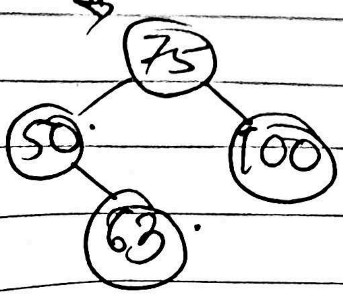
Delete 37 ✓



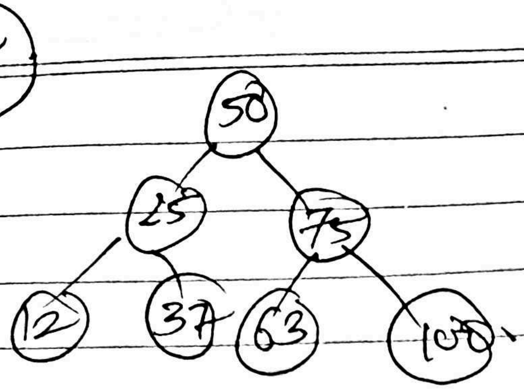
Delete: 25.



⇓



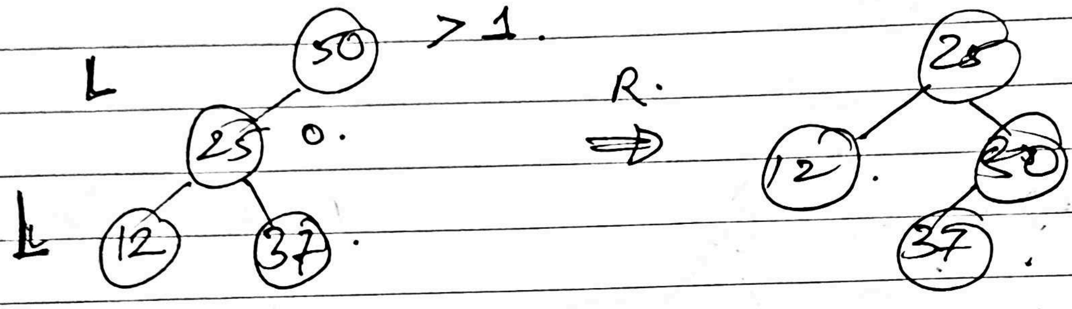
②



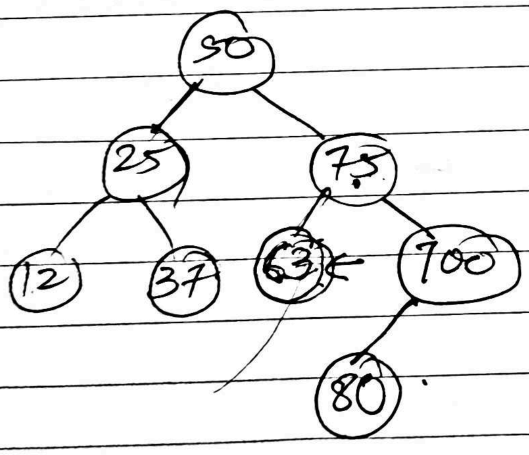
LL

BF(ROOT) > 1 AND BF(ROOT→LEFT) >= 0

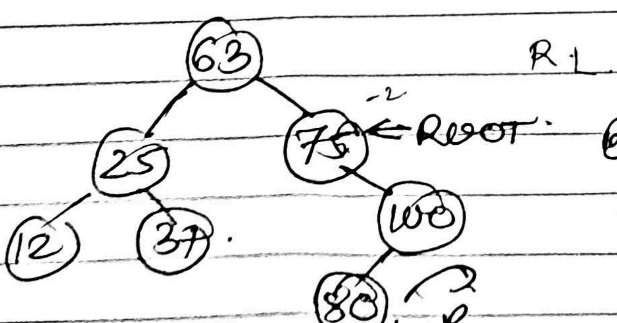
Delete. 100, 63, 75



③



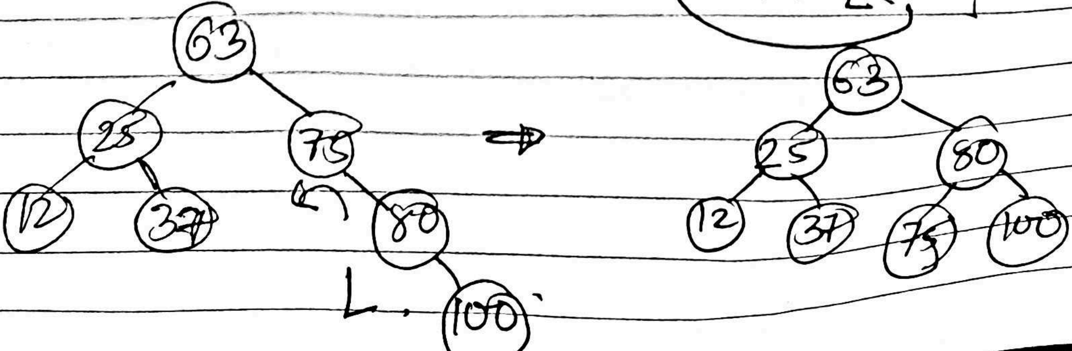
delete 50.



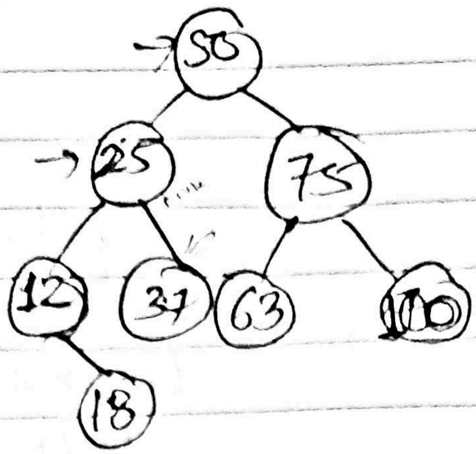
BF(ROOT) < -1

BF(ROOT→RIGHT) > 0.

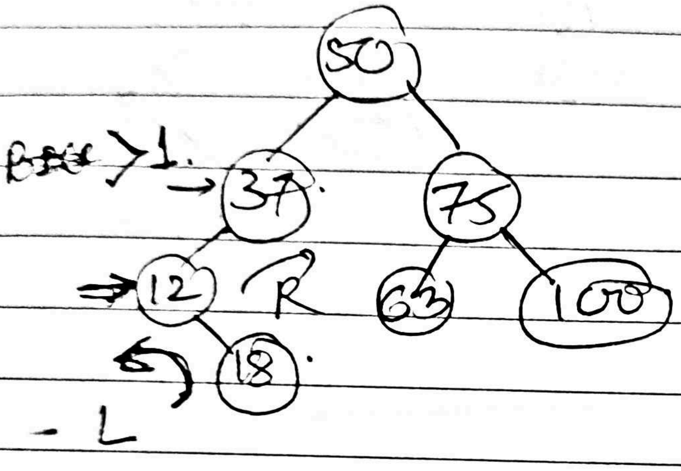
RL



(4).



Delete (25).



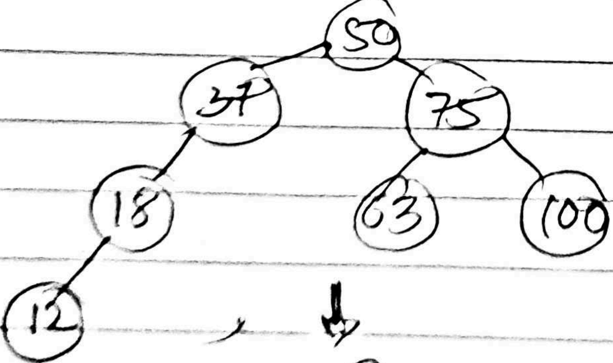
BF(ROOT) > 1.

L

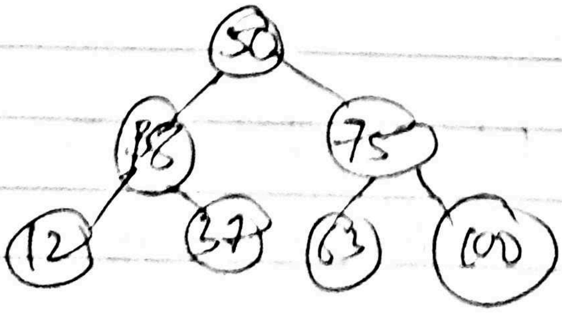
BF(ROOT -> LEFT < 0.

R.

LR.



50
37
75
18
63
100
12



50
18
75
12
37
63
100